МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
«БРАТСКИЙ ЦЕЛЛЮЛОЗНО – БУМАЖНЫЙ КОЛЛЕДЖ»
(ФГБПОУ «БЦБК»)

Специальность 09.02.07
Информационные системы и программирование

МЕТОДИЧЕСКОЕ ПОСОБИЕ

Курс лекций

по МДК 04.01 Внедрение и поддержка компьютерных систем

Братск, 2024

Составила (разработала) Юдина С.А., преподаватель кафедры информационных систем, программирования и автоматизации

Методическое пособие содержит полный курс лекций с вопросами по МДК 04.01 Внедрение и поддержка компьютерных систем. Данное пособие предназначено для студентов специальности 09.02.07 Информационные системы и программирование.

Рассмотрено на заседании кафедры информационных систем, программирования и автоматизации

«05» 04 2024г. № 8 _____

Одобрено и утверждено редакционным советом

_____

«22» 05 2024г. № 8

Рецензия

на методическое пособие «Курс лекций по МДК.04.01 Внедрение и поддержка компьютерных систем»
для специальности 09.02.07 Информационные системы и программирование
разработанные преподавателем кафедры ИСПиА Юдиной С.А.

Рабочая программа профессионального модуля разработана на основе Федерального государственного образовательного стандарта (далее – ФГОС) по специальности среднего профессионального образования (далее СПО) 09.02.07 «Информационные системы и программирование».

Рецензируемое учебное пособие содержит материалы, подготовленные автором для курса лекций по междисциплинарному курсу «Внедрение и поддержка компьютерных систем», читаемого обучающимся по специальности 09.02.07 Информационные системы и программирование.

Целью данного учебного пособия является систематическое изложение основных методов и средств эффективного анализа функционирования программного обеспечения.

Акцент в изложении материалов пособия сделан на изучении ключевых вопросов внедрения информационных систем: аппаратной и программной совместимости, тестировании в безопасном режиме, анализе журналов событий. Изучение этих разделов помогает обучающимся приобрести теоретические знания по изучаемому курсу.

В учебном пособии необходимый теоретический материал по курсу сопровождается большим количеством примеров, иллюстрирующих излагаемые разделы.

Считаю, что рецензируемое пособие может быть рекомендовано к опубликованию и использовано в учебном процессе.

Рецензент:
преподаватель кафедры ИСПиА
Ларева А.П.

Содержание

Введение

Внедрение программного обеспечения —это процесс настройки программного обеспечения под определенные условия использования, а также обучения пользователей работе с программным продуктом.

Одним из главных критериев успешности внедрения программного обеспечения - это выделение критических, с точки зрения общего результата, процедур в деятельности организации. Когда набор таких процедур определен, необходимо в первую очередь использовать программное обеспечение для автоматизации операций внутри именно этих процедур. Таким образом, разработанное программное обеспечение автоматически становится жизненно важным и востребованным для организации.

Сопровожде́ние (поддержка) программного обеспечения — процесс улучшения, оптимизации и устранения дефектов программного обеспечения (ПО) после передачи в эксплуатацию. Сопровождение ПО — это одна из фаз жизненного цикла программного обеспечения, следующая за фазой передачи ПО в эксплуатацию. В ходе сопровождения в программу вносятся изменения, с тем, чтобы исправить обнаруженные в процессе использования дефекты и недоработки, а также для добавления новой функциональности, с целью повысить удобство использования (юзабилити) и применимость ПО.

Сопровождение программного обеспечения стандартизовано, имеются национальные стандарты Российской Федерации, идентичные международным (ISO/IEC 12207:2008 System and software engineering — Software life cycle processes, ГОСТ Р ИСО/МЭК 12207-2010 «Национальный стандарт Российской Федерации. Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств»; ISO/IEC 14764:99 Information tehnology — Software maintenance, ГОСТ Р ИСО/МЭК 14764-2002 «Государственный стандарт Российской Федерации. Информационная технология. Сопровождение программных средств»; IEEE 1219).

1 Основные методы и средства эффективного анализа функционирования программного обеспечения

1.1 Информационные системы. Структура информационной системы

Система (от греческого *systema* — целое, составленное из частей соединение) — это совокупность элементов, взаимодействующих друг с другом, образующих определенную целостность, единство. Приведем некоторые понятия, часто использующиеся для характеристики системы.

1. *Элемент системы* — часть системы, имеющая определенное функциональное назначение. Сложные элементы систем, в свою очередь состоящие из более простых взаимосвязанных элементов, часто называют подсистемами.

2. *Организация системы* — внутренняя упорядоченность, согласованность взаимодействия элементов системы, проявляющаяся, в частности, в ограничении разнообразия состояний элементов в рамках системы.

3. *Структура системы* — состав, порядок и принципы взаимодействия элементов системы, определяющие основные свойства системы. Если отдельные элементы системы разнесены по разным уровням и внутренние связи между элементами организованы только от вышестоящих к нижестоящим уровням и наоборот, то говорят об *иерархической структуре* системы. Чисто иерархические структуры встречаются практически редко, поэтому, несколько расширяя это понятие, под иерархической структурой обычно понимают и такие структуры, где среди прочих связей иерархические связи имеют главенствующее значение.

4. *Архитектура системы* — совокупность свойств системы, существенных для пользователя.

5. *Целостность системы* — принципиальная несводимость свойств системы к сумме свойств отдельных ее элементов (эмерджентность свойств) и, в то же время, зависимость свойств каждого элемента от его места и функции внутри системы.

Информационная система — взаимосвязанная совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели»

В Федеральном законе «Об информации, информатизации и защите информации» дается следующее определение:

«Информационная система — организационно упорядоченная совокупность документов (массивов документов) и информационных технологий, в том числе с использованием средств вычислительной техники и связи, реализующих информационные процессы»

Структуру информационной системы составляет совокупность отдельных ее частей, называемых *подсистемами.*

Подсистема — это часть системы, выделенная по какому-либо признаку.

Общую структуру информационной системы можно рассматривать как совокупность подсистем независимо от сферы применения. В этом случае говорят о *структурном признаке* классификации, а подсистемы называют *обеспечивающими.* Таким образом, структура любой информационной системы может быть представлена совокупностью обеспечивающих подсистем (рисунок 1).

Рисунок 1 – Структура информационной системы как совокупность обеспечивающих подсистем

Среди обеспечивающих подсистем обычно выделяют *информационное*, *техническое*, *математическое*, *программное*, *организационное* и *правовое* обеспечение.

а) *Информационное обеспечение*

Назначение подсистемы информационного обеспечения состоит в своевременном формировании и выдаче достоверной информации для принятия управленческих решений.

Информационное обеспечение — совокупность единой системы классификации и кодирования информации, унифицированных систем документации, схем информационных потоков, циркулирующих в организации, а также методология построения баз данных.

Унифицированные системы документации создаются на государственном, республиканском, отраслевом и региональном уровнях. Главная цель — это обеспечение сопоставимости показателей различных сфер общественного производства. Разработаны стандарты, где устанавливаются требования:

• к унифицированным системам документации;

• к унифицированным формам документов различных уровней управления;

• к составу и структуре реквизитов и показателей;

• к порядку внедрения, ведения и регистрации унифицированных форм документов.

Однако, несмотря на существование унифицированной системы документации, при обследовании большинства организаций постоянно выявляется целый комплекс типичных недостатков:

• чрезвычайно большой объем документов для ручной обработки;

• одни и те же показатели часто дублируются в разных документах;

• работа с большим количеством документов отвлекает специалистов от решения непосредственных задач;

• имеются показатели, которые создаются, но не используются, и др.

Поэтому устранение указанных недостатков является одной из задач, стоящих при создании информационного обеспечения.

Схемы информационных потоков отражают маршруты движения информации и ее объемы, места возникновения первичной информации и использования результатной информации. За счет анализа структуры подобных схем можно выработать меры по совершенствованию всей системы управления.

Пример**.** В качестве примера простейшей схемы потоков данных можно привести схему, где отражены все этапы прохождения служебной записки или записи в базе данных о приеме на работу сотрудника — от момента ее создания до выхода приказа о его зачислении на работу.

Построение схем информационных потоков, позволяющих выявить объемы информации и провести ее детальный анализ, обеспечивает:

• исключение дублирующей и неиспользуемой информации;

• классификацию и рациональное представление информации.

При этом подробно должны рассматриваться вопросы взаимосвязи движения информации по уровням управления. Следует выявить, какие показатели необходимы для принятия управленческих решений, а какие нет. К каждому исполнителю должна поступать только та информация, которая используется.

Методология построения баз данных базируется на теоретических основах их проектирования. Для понимания концепции методологии приведем основные ее идеи в виде двух последовательно реализуемых на практике этапов:

1-й этап — обследование всех функциональных подразделений фирмы с целью:

• понять специфику и структуру ее деятельности;

• построить схему информационных потоков;

• проанализировать существующую систему документооборота;

• определить информационные объекты и соответствующий состав реквизитов (параметров, характеристик), описывающих их свойства и назначение.

2-й этап — построение концептуальной информационно-логической модели данных для обследованной на 1-м этапе сферы деятельности. В этой модели должны быть установлены и оптимизированы все связи между объектами и их реквизитами. Информационно-логическая модель является фундаментом, на котором будет создана база данных.

Для создания *информационного обеспечения* необходимо:

• ясное понимание целей, задач, функций всей системы управления организацией;

• выявление движения информации от момента возникновения и до ее использования на различных уровнях управления, представленной для анализа в виде схем информационных потоков;

• совершенствование системы документооборота;

• наличие и использование системы классификации и кодирования;

• владение методологией создания концептуальных информационно-

логических моделей, отражающих взаимосвязь информации;

• создание массивов информации на машинных носителях, что требует наличия современного технического обеспечения.

Техническое обеспечение — комплекс технических средств, предназначенных для работы информационной системы, а также соответствующая документация на эти средства и технологические процессы.

Комплекс технических средств составляют:

– компьютеры любых моделей;

– устройства сбора, накопления, обработки, передачи и вывода информации;

– устройства передачи данных и линий связи;

– оргтехника и устройства автоматического съема информации;

– эксплуатационные материалы и др.

Документацией оформляются предварительный выбор технических средств, организация их эксплуатации, технологический процесс обработки данных, технологическое оснащение. Документацию можно условно разделить на три группы:

• общесистемную, включающую государственные и отраслевые стандарты по техническому обеспечению;

• специализированную, содержащую комплекс методик по всем этапам разработки технического обеспечения;

• нормативно-справочную, используемую при выполнении расчетов по техническому обеспечению.

К настоящему времени сложились две основные формы организации технического обеспечения (формы использования технических средств): централизованная и частично или полностью децентрализованная.

Централизованное техническое обеспечение базируется на использовании в информационной системе больших ЭВМ и вычислительных центров.

Децентрализация технических средств предполагает реализацию функциональных подсистем на персональных компьютерах непосредственно на рабочих местах.

Перспективным подходом следует считать, по-видимому, *частично децентрализованный* подход — организацию технического обеспечения на базе распределенных сетей, состоящих из персональных компьютеров и большой ЭВМ для хранения баз данных, общих для любых функциональных подсистем.

Математическое и программное обеспечение — совокупность математических методов, моделей, алгоритмов и программ для реализации целей и задач информационной системы, а также нормального функционирования комплекса технических средств.

К средствам *математического обеспечения* относятся:

• средства моделирования процессов управления;

• типовые задачи управления;

• методы математического программирования, математической

статистики, теории массового обслуживания и др.

В состав *программного обеспечения* входят общесистемные и специальные программные продукты, а также техническая документация.

К *общесистемному* программному обеспечению относятся комплексы программ, ориентированных на пользователей и предназначенных для решения типовых задач обработки информации. Они служат для расширения функциональных возможностей компьютеров, контроля и управления процессом обработки данных.

Специальное программное обеспечение представляет собой совокупность программ, разработанных при создании конкретной информационной системы. В его состав входят пакеты прикладных программ, реализующие разработанные модели разной степени адекватности, отражающие функционирование реального объекта.

Техническая документация на разработку программных средств должна содержать описание задач, задание на алгоритмизацию, экономико-математическую модель задачи, контрольные примеры.

Организационное обеспечение — совокупность методов и средств, регламентирующих взаимодействие работников с техническими средствами и между собой в процессе разработки и эксплуатации информационной системы.

Организационное обеспечение реализует следующие функции:

- ✓ анализ существующей системы управления организацией, где будет использоваться ИС, и выявление задач, подлежащих автоматизации;
- ✓ подготовку задач к решению на компьютере, включая техническое задание на проектирование ИС и технико-экономическое обоснование ее эффективности;
- ✓ разработку управленческих решений по составу и структуре организации, методологии решения задач, направленных на повышение эффективности системы управления. Организационное обеспечение создается по результатам предпроектного обследования на 1-м этапе построения баз данных, с целями которого вы познакомились при рассмотрении информационного обеспечения.

Правовое обеспечение — совокупность правовых норм, определяющих создание, юридический статус и функционирование информационных систем, регламентирующих порядок получения, преобразования и использования информации.

Главной целью правового обеспечения является укрепление законности.

В состав правового обеспечения входят законы, указы, постановления государственных органов власти, приказы, инструкции и другие нормативные документы министерств, ведомств, организаций, местных органов власти. В правовом обеспечении можно выделить общую часть, регулирующую функционирование любой информационной системы, и локальную часть, регулирующую функционирование конкретной системы.

Правовое обеспечение этапов разработки информационной системы включает нормативные акты, связанные с договорными отношениями

разработчика и заказчика и правовым регулированием отклонений от договора.

Правовое обеспечение этапов функционирования информационной системы включает:

• статус информационной системы;
• права, обязанности и ответственность персонала;
• правовые положения отдельных видов процесса управления;
• порядок создания и использования информации и др.

1.2 Классификация ИС по различным признакам

Классификация по масштабу

По масштабу информационные системы подразделяются на следующие группы:

– одиночные;
– групповые;
– корпоративные.

Одиночные информационные системы реализуются, как правило, на автономном персональном компьютере (сеть не используется). Такая система может содержать несколько простых приложений, связанных общим информационным фондом, и рассчитана на работу одного пользователя или группы пользователей, разделяющих по времени одно рабочее место. Подобные приложения создайся с помощью так называемых настольных или локальных систем управления базами данных (СУБД). Среди локальных СУБД наиболее известными являются Clarion, Clipper, FoxPro, Paradox, dBase и Microsoft Access.

Групповые информационные системы ориентированы на коллективное использование информации членами рабочей группы и чаще всего строятся на базе локальной вычислительной сети. При разработке таких приложений используются серверы баз данных (Называемые также SQL-серверами) для рабочих групп. Существует довольно большое количество различных SQL-серверов, как коммерческих, так и свободно распространяемых. Среди них наиболее известны такие серверы баз данных, как Oracle, DB2, Microsoft SQL Server, InterBase, Sybase, Informix.

Корпоративные информационные системы являются развитием систем для рабочих групп, они ориентированы на крупные компании и могут поддерживать территориально разнесенные узлы или сети. В основном они имеют иерархическую структуру из нескольких уровней. Для таких систем характерна архитектура клиент-сервер со специализацией серверов или же многоуровневая архитектура. При разработке таких систем могут использоваться те же серверы баз данных, что и при разработке групповых информационных систем. Однако в крупных информационных системах наибольшее распространение получили серверы Oracle, DB2 и Microsoft SQL Server.

Для групповых и корпоративных систем существенно повышаются

требования к надежности функционирования и сохранности данных. Эти свойства обеспечиваются поддержкой целостности данных, ссылок и транзакций в серверах баз.

Классификация по сфере применения

По сфере применения информационные системы обычно подразделяются на четыре группы:

– системы обработки транзакций;

– системы принятия решений;

– информационно-справочные системы;

– офисные информационные системы.

Системы обработки транзакций, в свою очередь, по оперативности обработки данных, разделяются на пакетные информационные системы и оперативные информационные системы. В информационных системах организационного управлений преобладает режим оперативной обработки транзакций, для отражения *актуального* состояния предметной области в любой момент времени, а пакетная обработка занимает весьма ограниченную часть.

Системы поддержки принятия решений — DSS (Decision Support Systeq) — представляют собой другой тип информационных систем, в которых с помощью довольно сложных запросов производится отбор и анализ данных в различных разрезах: временных, географических и по другим показателям.

Обширный класс *информационно-справочных систем* основан на гипертекстовых документах и мультимедиа. Наибольшее развитие такие информационные системы получили в сети Интернет.

Класс *офисных информационных систем* нацелен на перевод бумажных документов в электронный вид, автоматизацию делопроизводства и управление документооборотом.

Классификация по способу организации

По способу организации групповые и корпоративные информационные системы подразделяются на следующие классы:

– системы на основе архитектуры файл-сервер;

– системы на основе архитектуры клиент-сервер;

– системы на основе многоуровневой архитектуры;

– системы на основе Интернет/интранет - технологий.

В любой информационной системе можно выделить необходимые функциональные компоненты, которые помогают понять ограничения различных архитектур информационных систем.

Архитектура файл-сервер только извлекает данные из файлов так, что дополнительные пользователи и приложения добавляют лишь незначительную нагрузку на центральный процессор. Каждый новый клиент добавляет вычислительную мощность к сети.

Архитектура клиент-сервер предназначена для разрешения проблем файл-серверных приложений путем разделения компонентов приложения и размещения их там, где они будут функционировать наиболее эффективно. Особенностью архитектуры клиент-сервер является использование

выделенных серверов баз данных, понимающих запросы на языке структурированных запросов SQL (Structured Query Language) и выполняющих поиск, сортировку и агрегирование информации.

В настоящее время архитектура клиент-сервер получила признание и широкое распространение как способ организации приложений для рабочих групп и информационных систем корпоративного уровня. Подобная организация работы повышает эффективность выполнения приложений за счет использования возможностей сервера БД, разгрузки сети и обеспечения контроля целостности данных.

Многоуровневая архитектура стала развитием архитектуры клиент-сервер и в своей классической форме состоит из трех уровней:

– нижний уровень представляет собой приложения клиентов, имеющие программный интерфейс для вызова приложения на среднем уровне;

– средний уровень представляет собой сервер приложений;

– верхний уровень представляет собой удаленный специализированный сервер базы данных.

Трехуровневая архитектура позволяет еще больше сбалансировать нагрузку на разные узлы и сеть, а также способствует специализации инструментов для разработки приложений и устраняет недостатки двухуровневой модели клиент-сервер.

В развитии *технологии Интернет/интранет* основной акцент пока что делается на разработке инструментальных программных средств. В то же время наблюдается отсутствие развитых средств разработки приложений, работающих с базами данных. Компромиссным решением для создания удобных и простых в использовании и сопровождении информационных систем, эффективно работающих с базами данных, стало объединение Интернет/интранет-технологии с многоуровневой архитектурой. При этом структура информационного приложения приобретает следующий вид: браузер — сервер приложений — сервер баз данных — сервер динамических страниц — web-сервер.

По характеру хранимой информации БД делятся на *фактографические* и *документальные*. Если проводить аналогию с описанными выше примерами информационных хранилищ, то фактографические БД — это картотеки, а документальные — это архивы. В фактографических БД хранится краткая информация в строго определенном формате. В документальных БД — всевозможные документы. Причем это могут быть не только текстовые документы, но и графика, видео и звук (мультимедиа).

Автоматизированная система управления (АСУ) - это комплекс технических и программных средств, совместно с организационными структурами (отдельными людьми пли коллективом), обеспечивающий управление объектом (комплексом) в производственной, научной или общественной среде.

Выделяют информационные системы управления образования (Например, кадры, абитуриент, студент, библиотечные программы). Автоматизированные системы для научных исследований (АСНИ), представляющие собой программно-аппаратные комплексы,

обрабатывающие данные, поступающие от различного рода экспериментальных установок и измерительных приборов, и на основе их анализа облегчающие обнаружение новых эффектов и закономерностей. Системы автоматизированного проектирования и геоинформационные системы.

Систему искусственного интеллекта, построенную на основе высококачественных специальных знании о некоторой предметной области (полученных от экспертов - специалистов этой области), называют экспертной системой. Экспертные системы - один из немногих видов систем искусственного интеллекта - получили широкое распространение, и нашли практическое применение. Существуют экспертные системы по военному делу, геологии, инженерному делу, информатике, космической технике, математике, медицине, метеорологии, промышленности, сельскому хозяйству, управлению, физике, химии, электронике, юриспруденции и т.д. И только то, что экспертные системы остаются весьма сложными, дорогими, а главное, узкоспециализированными программами, сдерживает их еще более широкое распространение.

Экспертные системы (ЭС) - это компьютерные программы, созданные для выполнения тех видов деятельности, которые под силу человеку-эксперту. Они работают таким образом, что имитируют образ действий человека-эксперта, и существенно отличаются от точных, хорошо аргументированных алгоритмов и не похожи на математические процедуры большинства традиционных разработок.

Классификация по степени автоматизации

В зависимости от степени автоматизации информационных процессов в системе управления фирмой информационные системы определяются как *ручные*, *автоматические*, *автоматизированные*.

Ручные ИС характеризуются отсутствием современных технических средств переработки информации и выполнением всех операций человеком. Например, о деятельности менеджера в фирме, где отсутствуют компьютеры, можно говорить, что он работает с ручной ИС.

Автоматические ИС выполняют все операции по переработке информации без участия человека.

Автоматизированные ИС предполагают участие в процессе обработки информации и человека, и технических средств, причем главная роль отводится компьютеру. В современном толковании в термин "информационная система" вкладывается обязательно понятие автоматизируемой системы.

Автоматизированные ИС, учитывая их широкое использование в организации процессов управления, имеют различные модификации и могут быть классифицированы, например, по характеру использования информации и по сфере применения.

Классификация по характеру использования информации:

Информационно-поисковые системы производят ввод, систематизацию, хранение, выдачу информации по запросу пользователя без сложных преобразований данных. Например, информационно-поисковая

система в библиотеке, в железнодорожных и авиа кассах продажи билетов.

Рисунок 2 – Классификация информационных систем по разным признакам

Информационно-решающие системы осуществляют все операции переработки информации по определенному алгоритму. Среди них можно провести классификацию по степени воздействия выработанной результатной информации на процесс принятия решений и выделить два класса: управляющие и советующие.

Управляющие ИС вырабатывают информацию, на основании которой человек принимает решение. Для этих систем характерен тип задач расчетного характера и обработка больших объемов данных. Примером могут служить система оперативного планирования выпуска продукции, система бухгалтерского учета.

Советующие ИС вырабатывают информацию, которая принимается человеком к сведению и не превращается немедленно в серию конкретных действий. Эти системы обладают более высокой степенью интеллекта, так как для них характерна обработка знаний, а не данных.

Классификация по сфере применения в организациях

Информационные системы организационного управления предназначены для автоматизации функций управленческого персонала. Учитывая наиболее широкое применение и разнообразие этого класса систем, часто любые информационные системы понимают именно в данном толковании. К этому классу относятся информационные системы управления как промышленными фирмами, так и непромышленными объектами: гостиницами, банками, торговыми фирмами и др.

Основными функциями подобных систем являются: оперативный контроль и регулирование, оперативный учет и анализ, перспективное и

оперативное планирование, бухгалтерский учет, управление сбытом и снабжением и другие экономические и организационные задачи.

ИС управления технологическими процессами (ТП) служат для автоматизации функций производственного персонала. Они широко используются при организации для поддержания технологического процесса в металлургической и машиностроительной промышленности.

ИС автоматизированного проектирования (САПР) предназначены для автоматизации функций инженеров-проектировщиков, конструкторов, архитекторов, дизайнеров при создании новой техники или технологии. Основными функциями подобных систем являются: инженерные расчеты, создание графической документации (чертежей, схем, планов), создание проектной документации, моделирование проектируемых объектов.

Интегрированные (корпоративные) ИС используются для автоматизации всех функций фирмы и охватывают весь цикл работ от проектирования до сбыта продукции. Создание таких систем весьма затруднительно, поскольку требует системного подхода с позиций главной цели, например получения прибыли, завоевания рынка сбыта и т.д. Такой подход может привести к существенным изменениям в самой структуре фирмы, на что может решиться не каждый управляющий.

Вопросы к Теме 1.1

1. Определения: «система», «элемент системы», «организация системы», «структура системы», «архитектура системы», «целостность системы», «подсистема»
2. Структура ИС (схема)
3. Виды подсистем
4. Методология построения баз данных
5. Классификация ИС по различным признакам

1.3 Принципы создания информационной системы

Многие пользователи компьютерной техники и программного обеспечения неоднократно сталкивались с ситуацией, когда программное обеспечение, хорошо работающее на одном компьютере, не работает на другом таком же устройстве. Или системные блоки одного вычислительного устройства не стыкуются с аппаратной частью другого. Или информационная система другой компании упорно не желает обрабатывать данные, которые вы подготовили в информационной системе у себя на рабочем месте. И так далее... Эта проблема называется проблемой совместимости вычислительных, телекоммуникационных и информационных устройств.

Развитие систем и средств вычислительной техники, расширенное их внедрение во все сферы науки, техники, сферы обслуживания и быта привели к необходимости объединения конкретных вычислительных устройств и реализованных на их основе информационных систем в *единые*

информационно-вычислительные системы (ИВС) и среды. При этом разработчики ИВС столкнулись с рядом проблем.

Например, *разнородность технических средств вычислительной техники* с точки зрения организации вычислительного процесса, архитектуры, системы команд, разрядности процессора и шины данных и т. д. потребовала создания *физических интерфейсов*, реализующих, как правило, взаимную совместимость устройств. При увеличении числа типов интегрируемых устройств сложность организации физического интерфейса между ними существенно возрастала. Разнородность программируемых сред, реализуемых в конкретных вычислительных устройствах и системах, с точки зрения многообразия операционных систем, различия в разрядности и прочих особенностей привела к созданию *программных интерфейсов* между устройствами и системами. При этом необходимо отметить, что достигнуть полной совместимости программных продуктов, разработанных для конкретной программной среды, в другой среде удавалось не всегда. Разнородность интерфейсов общения в системе "человек-компьютер" требовала постоянного согласования программно-аппаратного обеспечения и переобучения кадров.

1.3.1 Принцип «открытости» информационной системы

Решение проблем совместимости привело к разработке большого числа международных стандартов и соглашений в сфере применения информационных технологий и разработки информационных систем. Основополагающим понятием стало понятие ***открытые системы***.

Термин *открытая система* сегодня можно определить как "исчерпывающий и согласованный набор международных стандартов на информационные технологии и профили функциональных стандартов, которые специфицируют интерфейсы, службы и поддерживающие их форматы, чтобы обеспечить взаимодействие и мобильность программных приложений, данных и персонала".

Это определение, сформулированное специалистами института IEEE (Institute of Electrical and Electronic Engineers), унифицирует содержание среды, которую предоставляет открытая система для широкого использования. В настоящее время общепризнанным координационным центром по разработке и согласованию стандартов открытых систем является OASIS (Organization for the Advancement of Structured Information Standards).

Общие свойства *открытых информационных систем* можно сформулировать следующим образом:

- *расширяемость/масштабируемость* - обеспечение возможности добавления новых функций ИС или изменения некоторых уже имеющихся при неизменных остальных функциональных частях ИС;
- *мобильность/переносимость* - обеспечение возможности переноса программ и данных при модернизации или замене аппаратных платформ ИС и возможности работы с ними специалистов, пользующихся ИТ, без их переподготовки при изменениях ИС;
- *взаимодействие* - способность к взаимодействию с другими ИС

(технические средства, на которых реализована информационная система, объединяются сетью или сетями различного уровня - от локальной до глобальной);

• *стандартизуемость* - ИС проектируются и разрабатываются на основе согласованных международных стандартов и предложений, реализация открытости осуществляется на базе функциональных стандартов (профилей) в области информационных технологий;

• *дружественность к пользователю* - развитые унифицированные интерфейсы в процессах взаимодействия в системе "человек-машина" позволяют работать пользователю, не имеющему специальной "компьютерной" подготовки.

Новый взгляд на открытые системы определяется тем, что эти черты рассматриваются *в совокупности, как взаимосвязанные, и реализуются в комплексе*, что вполне естественно, поскольку все указанные выше свойства дополняют друг друга. Только в совокупности возможности открытых систем позволяют решать проблемы проектирования, разработки и внедрения современных информационных систем.

1.3.2 Структура среды информационной системы

Обобщенная структура любой ИС может быть представлена двумя взаимодействующими частями:

• функциональная часть, включающая прикладные программы, которые реализуют функции прикладной области;

• среда или системная часть, обеспечивающая исполнение прикладных программ.

С этим разделением тесно связаны две группы вопросов стандартизации:

• стандарты интерфейсов взаимодействия прикладных программ со средой ИС, *прикладной программный интерфейс* (Application Program Interface - API);

• стандарты интерфейсов взаимодействия самой ИС с внешней для нее средой (External Environment Interface - EEI).

Эти две группы интерфейсов определяют спецификации внешнего описания среды ИС - архитектуру, с точки зрения конечного пользователя, проектировщика ИС, прикладного программиста, разрабатывающего функциональные части ИС.

Спецификации внешних интерфейсов среды ИС и, как будет видно далее, спецификации интерфейсов взаимодействия между компонентами самой среды, - это точные описания всех необходимых функций, служб и форматов определенного интерфейса. Совокупность таких описаний составляет *эталонную модель открытых систем* (Reference Open System Model).

Эта модель используется более 20 лет и определяется системной сетевой архитектурой (SNA), предложенной IBM в 1974 году. Она основана на разбиении вычислительной среды на семь уровней, взаимодействие между которыми описывается соответствующими стандартами и обеспечивает связь

уровней *вне зависимости от построения уровня в каждой конкретной реализации* (рис. 3). Основным достоинством этой модели является детальное описание связей в среде с точки зрения технических устройств и коммуникационных взаимодействий. Вместе с тем она не принимает в расчет взаимосвязь с учетом мобильности прикладного программного обеспечения.

Рисунок 3 - Семиуровневая модель взаимодействия информационных систем

Эталонная модель среды открытых систем (OSE/RM) определяет разделение любой информационной системы на приложения (прикладные программы и программные комплексы) и среду, в которой эти приложения функционируют. Между приложениями и средой определяются стандартизованные интерфейсы (API), которые являются необходимой частью профилей любой открытой системы. Кроме того, в профилях ИС могут быть определены унифицированные интерфейсы взаимодействия функциональных частей друг с другом и интерфейсы взаимодействия между компонентами среды ИС.

1.4 Модель создания информационной системы

Методологически важно наряду с рассмотренными *моделями среды ИС* предложить *модель создания ИС*, которая имела бы те же аспекты функциональных групп компонентов (пользователи, функции, данные, коммуникации). Такой подход обеспечит сквозной процесс проектирования и сопровождения на всех стадиях эксплуатации ИС, а также возможность обоснованного выбора стандартов на разработку систем и документирование проектов.

Определение "компания" является сложной онтологической (понятийной) структурой, состоящей из определенной совокупности

сущностей и взаимосвязей (рис. 4). Взаимодействия между ее элементами, определяемые бизнес-логикой и закрепленные в наборе бизнес-правил, и являются деятельностью компании. Информационная система "отражает" логику и правила, организуя и преобразуя информационные потоки, автоматизирует процессы работы с данными и информацией и визуализирует результаты в виде наборов отчетных форм. Поэтому для начала следует создать бизнес-модель предприятия, являющуюся отображением предприятия и его информационно-управляющей системы. При создании модели формируется "язык общения" руководителей предприятия, консультантов, разработчиков и будущих пользователей, позволяющий выработать единое представление о том, ЧТО и КАК должна делать система управления предприятием (корпоративная система управления).

Рисунок 4 - Онтологическое поле современной компании

Такая бизнес-модель - осязаемый результат, с помощью которого можно максимально конкретизировать цели внедрения ИС и определиться со следующими параметрами проекта:

- основные цели бизнеса, которые можно достичь посредством автоматизации процессов;
- перечень участков и последовательность внедрения модулей ИС;
- фактическая потребность в объемах закупаемого программного и аппаратного обеспечения;
- реальные оценки сроков развертывания и запуска ИСУ;
- ключевые пользователи ИС и уточненный список членов команды внедрения;
- степень соответствия выбранного вами прикладного программного обеспечения специфике бизнеса вашей компании.

В основе модели всегда лежат бизнес-цели предприятия, полностью определяющие состав всех базовых компонентов модели:

- бизнес-функции, описывающие, ЧТО делает бизнес;
- основные, вспомогательные и управленческие процессы, описывающие, КАК предприятие выполняет свои бизнес-функции;
- организационно-функциональную структуру, определяющую, ГДЕ исполняются бизнес-функции и бизнес-процессы;
- фазы, определяющие, КОГДА (и в какой последовательности) должны быть внедрены те или иные бизнес-функции;
- роли, определяющие, КТО исполняет бизнес-функции и КТО является "хозяином" бизнес-процессов;
- правила, определяющие связь и взаимодействие между всеми ЧТО, КАК, ГДЕ, КОГДА и КТО.

После построения бизнес-модели (или параллельно с этим) можно приступать к формированию модели проектирования, реализации и внедрения самой ИС, как показано на рисунке 5.

Рисунок 5 – Модель проектирования, реализации и внедрения ИС

Опыт создания и использования "заказных" ИС позволяет условно выделить следующие основные этапы их жизненного цикла:

- *определение требований к системе и их анализ* - определение того, что должна делать система;
- *проектирование* - определение того, как система будет делать то, что она должна делать; проектирование - это прежде всего спецификация подсистем, функциональных компонентов и способов их взаимодействия в системе;
- *разработка* - создание функциональных компонентов и отдельных подсистем, соединение подсистем в единое целое;
- *тестирование* - проверка функционального соответствия системы показателям, определенным на этапе анализа;
- *внедрение* - установка и ввод системы в действие;
- *функционирование* - штатный процесс эксплуатации в соответствии с основными целями и задачами ИС;
- *сопровождение* - обеспечение штатного процесса эксплуатации

системы на предприятии заказчика.

Определение требований к системе и анализ являются первым этапом создания ИС, на котором требования заказчика уточняются, согласуются, формализуются и документируются. Фактически на этом этапе дается ответ на вопрос: "Для чего предназначена и что должна делать информационная система?". Именно здесь лежит ключ к успеху всего проекта.

Целью системного анализа является преобразование *общих, расплывчатых знаний* об исходной предметной области (требований заказчика) в *точные определения и спецификации* для разработчиков, а также генерация *функционального описания системы*. На этом этапе определяются и специфицируются:

- внешние и внутренние условия работы системы;
- функциональная структура системы;
- распределение функций между человеком и системой, интерфейсы;
- требования к техническим, информационным и программным компонентам системы;
- требования к качеству и безопасности;
- состав технической и пользовательской документации;
- условия внедрения и эксплуатации.

Разработка перечисленных выше спецификаций при создании ИС, предназначенной для автоматизации управленческих процессов, в общем случае проходит четыре стадии.

Первая стадия анализа - структурный анализ предприятия - начинается с исследования того, как организована система управления предприятием, с обследования функциональной и информационной структур системы управления, определения существующих и возможных потребителей информации.

По результатам обследования аналитик на первой стадии анализа выстраивает обобщенную логическую модель исходной предметной области, отображающую ее функциональную структуру, особенности основной деятельности и информационное пространство, в котором эта деятельность осуществляется (рис. 6). На этом материале аналитик строит функциональную модель "Как есть" (As Is).

Вторая стадия работы, к которой обязательно привлекаются *заинтересованные представители заказчика*, а при необходимости и независимые эксперты, состоит в анализе модели "*Как есть*", выявлении ее недостатков и узких мест, определении путей совершенствования системы управления на основе выделенных критериев качества.

Третья стадия анализа, содержащая элементы проектирования, - создание усовершенствованной обобщенной логической модели, отображающей реорганизованную предметную область или ее часть, которая подлежит автоматизации - модель "*Как должно быть*" (As To Be).

Заканчивается процесс (четвертая стадия) разработкой "Карты автоматизации", представляющей собой модель реорганизованной предметной области, на которой *обязательно обозначены "границы*

автоматизации".

В большинстве случаев модель "*Как есть*" улучшается системным аналитиком за счет устранения очевидных несоответствий и узких мест, а полученный таким образом вариант модели рассматривается в дальнейшем в качестве предварительной модели "*Как должно быть*", которая впоследствии дополняется в соответствии со стратегией развития предприятия, как представлено на рисунке 7.

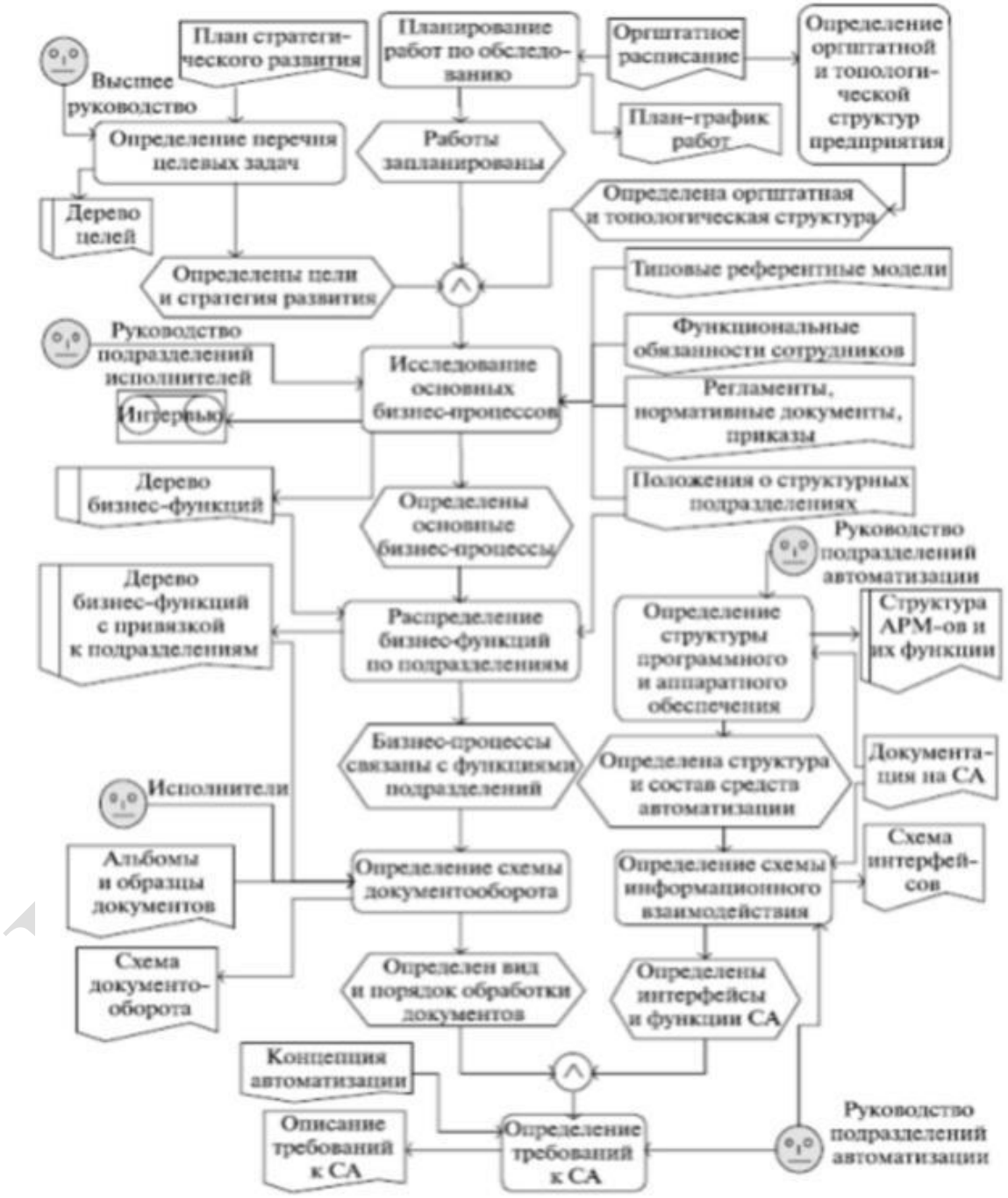


Рисунок 6 - Схема обследования предприятия

Рисунок 7 - Стадии построения модели информационной системы

На *стадии анализа требований* к проектируемой системе вводятся:

- классы пользователей и соответствующие диаграммы бизнес-транзакций;
- модели (диаграммы) процессов прикладной деятельности и соответствующие перечни функциональных задач ИС;
- классы объектов предметной области и соответствующие диаграммы "сущность-связь", отражающие информационную модель этой предметной области;
- топология расположения подразделений и пользователей, обслуживаемых данной ИС;
- параметры защиты данных, информации и самой системы.

Основным документом, отражающим результаты работ первого этапа создания ИС, является *техническое задание на проект (разработку)*, содержащее, кроме вышеперечисленных определений и спецификаций, также сведения об очередности создания системы, сведения о выделяемых ресурсах, директивных сроках проведения отдельных этапов работы, организационных процедурах и мероприятиях по приемке этапов, защите проектной информации и т.д.

Следующий этап - *проектирование*. В реальных условиях проектирование - это поиск, моделирование способа разработки, который удовлетворяет требованиям функциональности системы средствами имеющихся технологий с учетом заданных начальных условий и ограничений. Проектирование информационных систем всегда начинается с определения цели проекта. Основная задача любого успешного проекта заключается в том, чтобы на момент запуска системы и в течение всего времени ее эксплуатации можно было обеспечить:

- требуемую функциональность системы и степень адаптации к изменяющимся условиям ее функционирования;
- требуемую пропускную способность системы и минимальное время реакции системы на запрос;
- безотказную работу системы в требуемом режиме, готовность и доступность системы для обработки запросов пользователей;
- простоту эксплуатации и сопровождения системы;

• необходимую безопасность данных и права доступа пользователей.

Производительность и надежность являются главными факторами, определяющими эффективность системы. Хорошее проектное решение служит основой высокопроизводительной системы.

Проектирование информационных систем охватывает три основные области:

• проектирование структур данных, которые будут реализованы в базе данных;

• проектирование программ, экранных форм, отчетов, которые будут обеспечивать выполнение запросов к данным;

• проектирование конкретной среды или технологии, а именно: топологии сети, конфигурации аппаратных средств, используемой архитектуры, параллельной обработки, распределенной обработки данных и т.п.

На основе результатов системного анализа *на стадии предварительного проекта* разрабатываются:

• проект программно-аппаратной реализации, проект пользовательских интерфейсов и технологии работы пользователей в системе;

• архитектура распределенной системы и спецификации телекоммуникационной сети;

• модели (диаграммы) потоков данных;

• функциональные блок-схемы прикладного и системного программного обеспечения (последние - в соответствии с принятыми моделями среды ИС и профилями стандартов).

Стадия предварительного проекта может предусматривать прототипирование фрагментов, важных с точки зрения пользователя, для проверки их соответствия требованиям на ранней фазе разработки.

На стадии детального проектирования разрабатываются:

• комплексы функциональных программ ИС и проект реализации среды ИС;

• структуры данных, средства ведения баз данных;

• сетевые адреса, протоколы телекоммуникаций и другие компоненты среды обмена информацией, включаемые в состав проектируемой ИС;

• правила разграничения доступа пользователей и средства их реализации.

Стадия реализации ИС предусматривает разработку и тестирование компонентов и комплексное тестирование системы.

Стадия эксплуатации и сопровождения предусматривает контроль функционирования, внесение требуемых изменений в информационную базу в процессе текущей работы и модернизацию функций ИС силами прикладных специалистов с помощью инструментальных средств, встроенных в систему.

Этапы разработки, тестирования, внедрения, эксплуатации и сопровождения ИС объединяются термином реализация. Реализация ИС является чрезвычайно сложным многоаспектным процессом, осуществляемым на базе совокупностей (профилей) гармонизированных

международных стандартов, спецификаций и соглашений. Такая практика является залогом того, что создаваемая информационная система будет реализована как "открытая система". Иными словами, такая ИС будет масштабируема, мобильна, переносима, обладать дружественными интерфейсами и т.д.

Жизненный цикл ИС формируется в соответствии с принципом нисходящего проектирования и, как правило, носит спирально-итерационный характер. Реализованные этапы, начиная с самых ранних, циклически повторяются в соответствии с изменениями требований и внешних условий, введением дополнительных ограничений и т.п. На каждом этапе жизненного цикла порождается определенный набор технических решений и документов, при этом для каждого этапа исходными являются документы и решения, принятые на предыдущем этапе. Жизненный цикл ИС заканчивается, когда прекращается ее программное и техническое сопровождение.

1.5 Реинжиниринг бизнес-процессов

Внедрение информационных технологий и реализованных на их основе информационных систем в повседневную деятельность предприятия дает ему тактические и долгосрочные преимущества в бизнесе. Стремление руководства к использованию ИТ может остаться лишь благими намерениями, если оно не будет следовать сложившимся требованиям и правилам разработки, проектирования и внедрения ИТ. Выше говорилось о базовых требованиях к стандартизации объектов и функциональных задач, без которых реализуемая система не будет являться *открытой системой*, что приведет впоследствии к многочисленным проблемам при ее внедрении и эксплуатации.

Следование требованиям стандартов при разработке ИС автоматически приводит к тому, чтобы само предприятие - внешняя среда для ИС - также отвечало необходимым требованиям: определение и стандартизация классов пользователей и объектов, топология потоков данных и работ, архитектура наследуемых систем, состояние бизнес-процессов и т.д.

Бизнес-процесс представляет собой систему последовательных, целенаправленных и регламентированных видов деятельности, в которой посредством управляющего воздействия и с помощью определенных ресурсов за определенное время входы процесса преобразуются в выходы - в результаты, представляющие ценность для потребителя и приносящие прибыль изготовителю.

Бизнес-процесс в масштабах предприятия реализуется в виде сети основных, вспомогательных, поддерживающих и управленческих процессов (рис. 8).

Рисунок 8 – Сеть основных, вспомогательных, поддерживающих и управленческих процессов

При этом разделение на основные и вспомогательные процессы в определяющей степени зависит от предметной области и направления деятельности предприятия: для производственной компании, например, деятельность юридического отдела является вспомогательной, а для юридической или консалтинговой фирмы - основной. Идентификация процессов является обязательным условием, без реализации которого невозможна информатизация деятельности.

Руководители предприятия, решившиеся на внедрение ИТ, должны твердо усвоить: начало работ по проектированию информационной системы чаще всего влечет за собой обязательный реинжиниринг бизнес-процессов! Реинжиниринг представляет собой множество методик и рекомендаций, среди них нужно выбрать те, которые наилучшим образом удовлетворяют поставленным целям.

Реинжиниринг бизнес-процессов - это совокупность методов и действий, служащих для перепроектирования процессов в соответствии с изменившимися условиями внешней и внутренней среды и/или целями бизнеса.

Существует несколько базовых правил, которых следует придерживаться в процессе проведения реинжиниринга:

- разработка последовательных пошаговых процедур для перепроектирования процессов;
- использование в проектировании стандартных языков и нотаций;
- наличие эвристических и прагматических показателей, позволяющих оценить или измерить степень соответствия перепроектированного процесса или функциональности заданным целям;
- подход к решению частных задач и к их совокупности должен быть

системным;

• даже небольшое улучшение должно давать быстрый положительный эффект.

Реинжиниринг деловых процессов и функций начинается с пересмотра целей предприятия, его структуры, анализа потребностей внутренних пользователей и рынка, производимых продуктов и услуг (рис. 9).

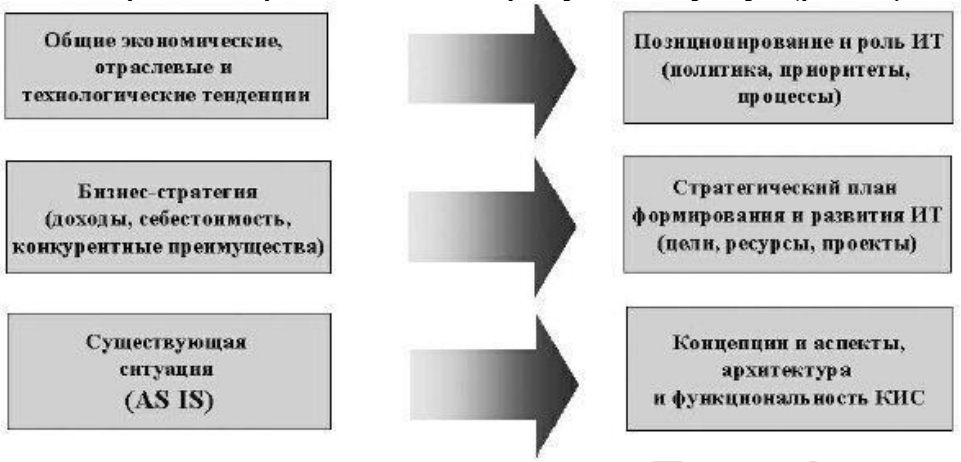


Рисунок 9 – Пересмотр целей предприятия

Перепланирование целей и задач предполагает пересмотр политики предприятия и ответа на следующие вопросы:

• Какие новые вызовы предъявляют нам изменившиеся условия бизнеса?

• Что представляет собой предприятие сейчас, и что мы хотим от него в будущем?

• Каких именно потребителей мы обслуживаем, насколько мы удовлетворяем их требования и ожидания, и что нужно сделать для привлечения новых?

• Какие именно показатели определяют эффективность деятельности предприятия, производительность труда и качество продукта, является ли это определение полным и адекватным?

• Какие именно информационные технологии и средства помогут нам в этом?

Для ответа на эти ключевые вопросы необходимо в первую очередь провести детальное описание *бизнес-архитектуры предприятия, его бизнес-логики*, построить *функциональную модель* взаимодействия бизнес-процессов, ресурсов и персонала и отразить ее в архитектуре ИС, содержании модулей информационных подсистем и визуализации форм представления информации. Необходимо также иметь методики и инструменты реорганизации процессов, решения прикладных задач и управления проектом реинжиниринга (рис. 10).

Описание бизнес-архитектуры позволяет:

• построить схему основных потоков данных, работ, движения финансов и документов;

• понять, как информация распределяется между подразделениями и кто является конечным пользователем в том или ином бизнес-процессе;

• описать взаимодействие процессов и модулей информационной

системы;

• определить критическую важность видов информации для конкретных уровней управления предприятием;

• выявить дублированные структуры и связи.

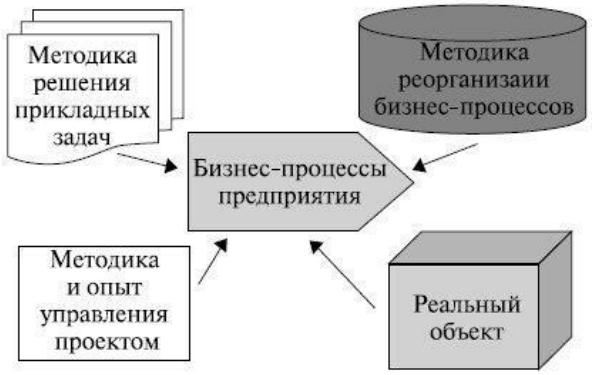


Рисунок 10 - Базовая основа улучшения процесса

Результатом такого описания является:

• уточненная карта сети процессов;

• матрица взаимосвязей процессов и подразделений, вовлеченных в эти процессы;

• информация о том, какие системы автоматизации существуют, при выполнении каких операций применяются, где и какие данные используются, какие системы автоматизации и информатизации необходимо разработать;

• функциональные схемы потоков данных (Data Flow), работ (Work Flow), финансовых потоков (Cash Flow), потоков управленческих воздействий (Control Flow) и документооборота (Doc Flow).

Функциональная модель поможет составить точные спецификации всех операций, процедур и взаимосвязей между ними. Такая модель, если она построена правильно, обеспечивает исчерпывающее описание функционирующего процесса и всех имеющихся в нем потоков информации. Эта модель описывает состояние "*Как есть*" (As Is). По результатам анализа возможных путей улучшения от реальной модели нужно перейти к модели, характеризующей улучшения: модель "*Как будет*" (As To Be), вариант "*Как должно быть*" (рис. 11).

Рисунок 11 - Схема реинжиниринга бизнес-процесса

Функциональное моделирование является достаточно серьезной

проблемой. Его полнота и соответствие построенной модели зависят как от средств моделирования, так и от квалификации специалистов, выполняющих это моделирование.

Реинжиниринг бизнес-процессов является сложным и многоаспектным проектом, требующим тщательного планирования и проработки деталей. В таблице1 показаны основные этапы реинжиниринга.

Таблица 1 – Основные этапы реинжиниринга

Этап	**Мероприятия**
Планирование и начало работ	Выявление главных причин проведения реформы на предприятии и оценка последствий отказа от такой реформы
	Выявление важнейших процессов, требующих реинжиниринга
	Выявление единомышленников среди руководства и создание рабочей группы из представителей администрации
	Обеспечение поддержки проекта руководством
	Подготовка плана проекта: определение объема, обозначение измеримых целей, выбор методологии, составление подробного графика
	Согласование целей и объемов проекта с руководством
	Формирование группы реинжиниринга
	Выбор консультантов или внешних экспертов
	Проведение вводного совещания
	Доведение целей проекта до руководителей низшего звена; начальное информирование всей организации
	Обучение группы реинжиниринга
	Подготовка плана и начало работ
Исследования	Аналитическое исследование опыта компаний с подобными процессами
	Опрос клиентов и контрольных групп для выявления существующих и будущих требований
	Опрос служащих и руководителей для выявления вопросов; мозговой штурм
	Поиск в литературе и прессе данных о тенденциях в отрасли и о чужом опыте
	Оформление подробных документов на исходные процессы и сбор рабочих данных; выявление недоработок
	Обзор изменений и вариантов технологий
	Опрос владельцев и представителей руководства

	Посещение кружков и семинаров
	Сбор данных от внешних экспертов и консультантов
Проектирование	Мозговой штурм и выработка новаторских идей; упражнения по творческому мышлению, чтобы "снять шоры"
	Проработка сценариев "а что, если?" и применение "шаблонов успеха" других компаний
	Создание при помощи специалистов 3-5 моделей; разработка комплексных моделей, в которых собрано лучшее от каждой из предыдущих
	Создание картины идеального процесса
	Определение моделей нового процесса и их графическое представление
	Разработка организационной модели в сочетании с новым процессом
	Определение технологических требований; выбор платформы для новых процессов
	Выделение краткосрочных и долгосрочных мер
Утверждение	Анализ затрат и преимуществ; расчет прибыли на капитал
	Оценка влияния на клиентов и служащих; оценка влияния на конкурентоспособность
	Подготовка официального документа для высшего руководства
	Проведение обзорных совещаний для ознакомления и утверждения деталей проекта оргкомитетом и высшим руководством
Внедрение	Завершение подробной разработки процессов и организационных моделей; определение новых рабочих обязанностей
	Разработка систем поддержки
	Реализация предварительных вариантов и первичные испытания
	Ознакомление работников с новым вариантом; разработка и осуществление плана реформы
	Разработка поэтапного плана; внедрение как таковое
	Разработка плана обучения; обучение работников новым процессам и системам
Последующие мероприятия	Разработка мероприятий по периодической оценке; определение итогов нового процесса; внедрение программы непрерывного совершенствования нового

	процесса
	Предоставление окончательного отчета оргкомитету и администрации

1.6 Отображение и моделирование процессов

На сегодняшний день получили распространение три основные методологии функционального моделирования (и сопутствующий им инструментарий): IDEF (Integrated DEFinition), UML (Unified Modeling Language) и ARIS (Architecture of Integrated Information Systems). Для каждой из них существуют определенные программные продукты, которые помимо разработки позволяют проводить преобразования и операции для последующей работы с полученными моделями. Наибольшее распространение сегодня получили методологии IDEF и программные продукты BPWin, содержащие методологии IDEF0, IDEF3, DFD (Data Flow Diagrams) и ERWin (IDEF1x) от компании Computer Associates.

История IDEF начинается с 70-х годов XX века с методологии SADT (Structured Analysis and Design Technique), разработанной Дугласом Россом (Softtech INC). Изначально SADT применялось Министерством обороны США для практического моделирования процессов в рамках программы ICAM (Integrated Computer Aided Manufacturing). Принципиальным требованием при разработке рассматриваемого семейства методологий была возможность эффективного обмена информацией между всеми специалистами - участниками программы ICAM (Icam DEFinition). В последующем эта методология была трансформирована в стандарт IDEF0 (Function Modeling, FIPS №183). Семейство IDEF включает уже упомянутые IDEF3 (Process Description Capture) и IDEF1x (Data Modeling, FIPS №184).

После опубликования стандарты были успешно применены в самых различных областях бизнеса, показав себя эффективным средством *анализа, конструирования и отображения бизнес-процессов* (к слову сказать, они активно применяется и в отечественных госструктурах, например, в Государственной Налоговой Инспекции). Более того, собственно с широким применением IDEF (и предшествующей методологии SADT) и связано возникновение основных идей популярного ныне понятия "реинжиниринг бизнес-процессов" (Business Process Reengineering - BPR).

Информационный процесс - это устойчивый процесс (последовательность работ и действий с данными и информацией), относящийся к сопровождению производственно-хозяйственной деятельности компании и обычно ориентированный на информационное обслуживание создания новой стоимости. Бизнес-процесс включает в себя иерархию взаимосвязанных функциональных действий, реализующих одну (или несколько) бизнес-целей компании и отражающий результаты в информационной системе, например, информационное обеспечение управления и анализа выпуска продукции или ресурсное обеспечение

выпуска продукции (под продукцией здесь понимают товары, услуги, решения, документы).

Работа с использованием метода IDEF начинается с постановки цели моделирования. Мировой опыт свидетельствует, что *ошибки при постановке цели* приводят в среднем к 50% неудач в процессе моделирования. Формулирование цели изначально направляет работу в заданном направлении, а значит, ограничивает круг вопросов для анализа. Практическая работа начинается с определения контекста (Context, Context Diagram), то есть верхнего уровня системы, в нашем случае - предприятия. После формулировки цели необходимо очертить область моделирования (Scope), которая в последующем будет определять общие направления движения и глубину детализации (Decomposition). Собственно, сама методология IDEF определяет стандартизированные объекты для работы и отображения. Например, к таковым относятся функция (Activity), интерфейсная дуга (Arrow), заметка (Note), а также способ их расположения и трактования (Semantics).

В последнее время на российском рынке появился программный продукт *Business Studio*, который специально создан для работы с методами IDEF и обладает интуитивным и дружественным интерфейсом (User-friendly Interface).

В основе нотации и методологии IDEF0 лежит понятие "*блока*", то есть прямоугольника, который выражает некоторую функцию бизнеса (рис. 12). В соответствии со стандартом функция должна быть выражена глагольным оборотом. В IDEF0 роли сторон прямоугольника (функциональные значения) различны: верхняя сторона имеет значение "*управление*", левая - "*вход*", правая - "*выход*", нижняя - "*механизм исполнения*".

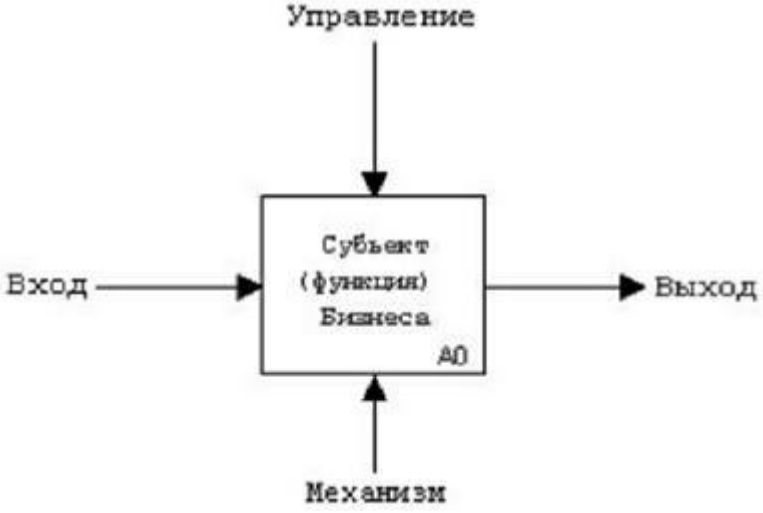


Рисунок 12 - Базовый блок методологии IDEF0

Вторым элементом методологии и нотации является "*поток*", называемый в стандарте "*интерфейсная дуга*". Это элемент, описывающий данные, неформальное управление или что-либо другое, - то, что оказывает влияние на функцию, изображенную блоком. Потоки обозначаются *оборотом существительного*.

В зависимости от того, к какой стороне блока направлен поток, он, соответственно, носит название "входной", "выходной" или "управляющий".

Изобразительным элементом, представляющим поток, является стрелка. Поток можно интерпретировать как *представление объекта*, под которым понимается как информационный объект, так и реальный физический объект.

Важным фактором является то, что "*источником*" и "*приемником*" потоков (то есть началом и концом стрелки) могут быть, как правило, только блоки. При этом источником может являться только выходная сторона блока, приемником - любая из трех оставшихся. Если же необходимо подчеркнуть внешний характер потока, то может быть применен метод "туннелирования" - скрытие или появление интерфейсной дуги из "туннеля".

И, наконец, "третьим китом" методологии IDEF0 является принцип *функциональной декомпозиции блоков*, который представляет собой модельную интерпретацию той практической ситуации, что любое действие (тем более такое сложное, как бизнес-процесс) может быть разбито (декомпозировано) на более простые операции (действия, бизнес- функции). Или, другими словами, действие может быть представлено как совокупность элементарных функций.

Пример функциональной модели процесса отгрузки и доставки продукции показан на рис. 13.

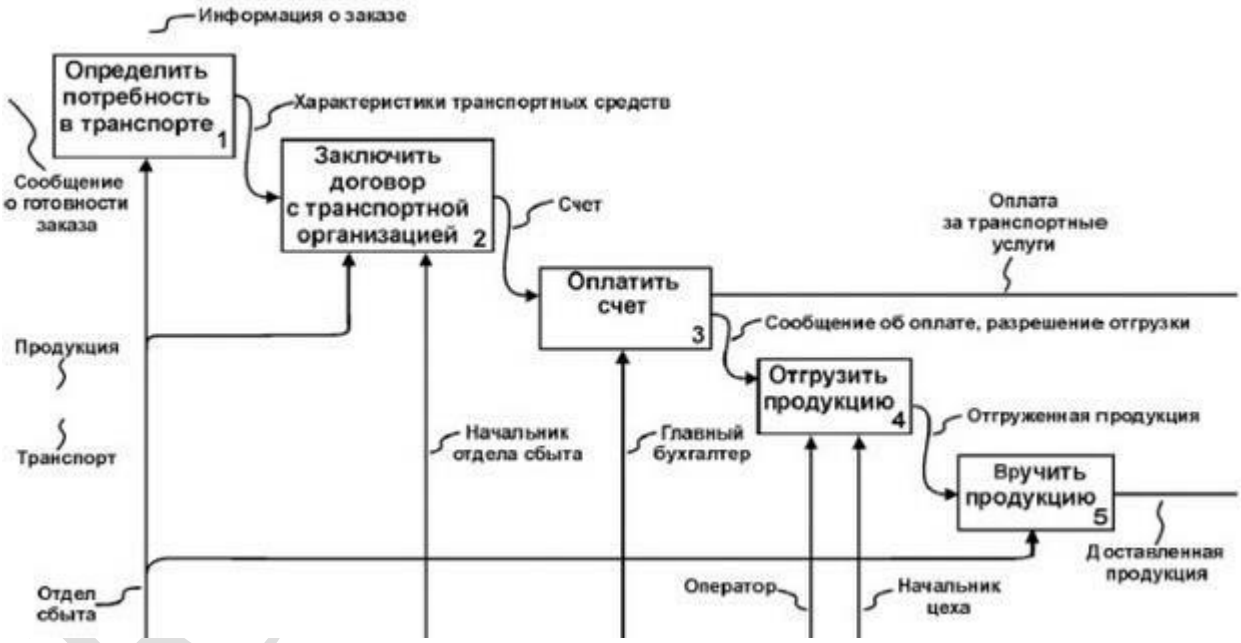


Рисунок 13 - Пример функциональной модели процесса отгрузки и доставки

1.7 Обеспечение процесса анализа и проектирования ИС возможностями CASE-технологий

Термин "**CASE**" (Computer Aided Software/System Engineering) используется в настоящее время в весьма широком смысле. Первоначальное значение термина "CASE", ограниченное вопросами автоматизации разработки только лишь программного обеспечения (ПО), в настоящее время приобрело новый смысл, охватывающий процесс разработки сложных ИС в целом.

Теперь под термином "CASE-средства" понимаются программные

средства, поддерживающие процессы создания и сопровождения ИС, включая анализ и формулировку требований, проектирование прикладного программного обеспечения (приложений) и баз данных, генерацию кода, тестирование, документирование, обеспечение качества, конфигурационное управление и управление проектом, а также другие процессы.

Появлению CASE-технологии и CASE-средств предшествовали исследования в области методологии программирования. Программирование обрело черты системного подхода с разработкой и внедрением языков высокого уровня, методов структурного и модульного программирования, средств визуального моделирования и проектирования на базе языка UML (Unified Modeling Language), средств их поддержки, формальных и неформальных языков описаний системных требований и спецификаций и т.д. Кроме того, появлению CASE-технологии способствовали и такие факторы, как:

- подготовка аналитиков и программистов, восприимчивых к концепциям модульного и структурного программирования;
- широкое внедрение и постоянный рост производительности компьютеров, позволившие использовать эффективные графические средства и автоматизировать большинство этапов проектирования;
- внедрение сетевой технологии, которая предоставила возможность объединения усилий отдельных исполнителей в единый процесс проектирования путем использования разделяемой базы данных, содержащей необходимую информацию о проекте.

CASE-технология представляет собой методологию проектирования ИС, а также набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех этапах разработки и сопровождения ИС и разрабатывать приложения в соответствии с информационными потребностями пользователей. Большинство существующих CASE-средств основано на методологиях структурного (в основном) или объектно-ориентированного анализа и проектирования, использующих спецификации в виде диаграмм или текстов для описания внешних требований, связей между моделями системы, динамики поведения системы и архитектуры программных средств [Вендров А.М., www.citforum.ru/database/case/index.shtml].

CASE-средства позволяют создавать не только продукт, практически готовый к применению, но и обеспечить "правильный" процесс его разработки. Основная цель технологии - отделить проектирование программного обеспечения от его кодирования, сборки, тестирования и максимально "скрыть" от будущих пользователей все детали разработки и функционирования ПО. При этом значительно повышается эффективность работы проектировщика: сокращается время разработки, уменьшается число программных ошибок, программные модули можно использовать при следующих разработках.

Большинство CASE-средств основано на парадигме "методология/метод/нотация/структура/средство".

Методология задает руководящие указания для оценки и выбора

проекта разработки ПО, этапы и последовательность работ, правила применения тех или иных методов.

Метод - систематическая процедура или технология генерации описаний компонент ПО (например, описание потоков и структур данных).

Нотации предназначены для описания системы в целом, ее элементов, таких как графы, диаграммы, таблица, блок-схемы, алгоритмы, формальные языки и языки программирования.

Структуры являются средством для реализации структурного анализа и построения структуры конкретной системы.

Средства - технологические и программные инструменты для поддержки и усиления методов.

CASE-технологии обладают следующими основными достоинствами, которые позволяют широко использовать их при разработке информационных систем:

- ускоряют процесс коллективного проектирования и разработки;
- позволяют за короткий срок создать прототип заказанной системы с заданными свойствами;
- освобождают разработчика от рутинной работы, оставляя время для творчества;
- обеспечивают эффективность и качество разрабатываемого ПО за счет автоматизации контроля всего процесса разработки;
- поддерживают сопровождение и развитие системы на высоком уровне.

Следует отметить, что, несмотря на все потенциальные возможности CASE-средств, существует достаточно много примеров их неудачного внедрения, в результате которых CASE-средства становятся "полочным" ПО (Shelfware).

В связи с этим необходимо учитывать следующее:

- CASE-средства не обязательно дают немедленный эффект, он может быть получен только спустя какое-то время;
- реальные затраты на внедрение CASE-средств обычно намного превышают затраты на их приобретение;
- CASE-средства обеспечивают возможности для получения существенной выгоды только после успешного завершения процесса их внедрения, эффективного обучения пользователей и регулярного применения.

Можно также перечислить следующие факторы, усложняющие определение возможного эффекта от использования CASE-средств:

- широкое разнообразие качества и возможностей CASE-средств;
- относительно небольшое время использования CASE-средств в различных организациях и недостаток опыта их применения;
- широкое разнообразие в практике внедрения различных организаций;
- отсутствие детальных метрик и данных для уже выполненных и текущих проектов;
- широкий диапазон предметных областей проектов;
- различная степень интеграции CASE-средств в различных проектах.

Некоторые аналитики считают, что реальная выгода от использования некоторых типов CASE-средств может быть получена только после одно- или двухлетнего опыта. Другие полагают, что воздействие может реально проявиться в фазе эксплуатации жизненного цикла ИС, когда технологические улучшения могут привести к снижению эксплуатационных затрат.

Ниже перечислены основные виды и последовательность работ, рекомендуемые при построении логических моделей предметной области в рамках CASE-технологии анализа системы управления предприятием:

1. *Проведение функционального и информационного обследования системы управления* (административно-управленческой деятельности) предприятием, как показано на рисунке 14:

- определение организационно-штатной структуры предприятия;
- определение функциональной структуры предприятия;
- определение перечня целевых функций структурных элементов (подразделений и должностных лиц);
- определение круга и очередности обследования структурных элементов системы управления согласно сформулированным целевым функциям;
- обследование деятельности выделенных структурных элементов;
- построение FD-диаграммы системы управления с указанием структурных элементов и функций, реализация которых будет моделироваться на DFD-уровне.

Рисунок 14 - Модель системы в технологическом CASE-решении

2. *Разработка моделей деятельности структурных элементов и системы управления в целом*:

- выделение множества внешних объектов, оказывающих существенное влияние на деятельность структурного элемента;

- спецификация входных и выходных информационных потоков;
- выявление основных процессов, определяющих деятельность структурного элемента и обеспечивающих реализацию его целевых функций;
- спецификация информационных потоков между основными процессами деятельности, уточнение связей между процессами и внешними объектами;
- оценка объемов, интенсивности и других необходимых характеристик информационных потоков;
- разработка иерархии диаграмм потоков данных, образующих функциональную модель деятельности структурного элемента;
- объединение DFD-моделей структурных элементов в единую модель системы управления предприятием.

3. *Разработка информационных моделей структурных элементов и модели информационного пространства системы управления*:

- определение сущностей модели и их атрибутов;
- проведение атрибутного анализа и оптимизация сущностей;
- идентификация отношений между сущностями и определение типов отношений;
- анализ и оптимизация информационной модели;
- объединение информационных моделей в единую модель информационного пространства.

4. *Разработка предложений по автоматизации системы управления предприятием*:

- определение границ автоматизации - составление перечня автоматизируемых структурных элементов, разбиение процессов основной деятельности на автоматические, автоматизированные и ручные;
- составление перечня подсистем и логических АРМов (автоматизированных рабочих мест), определение способов их взаимодействия;
- разработка предложений по очередности проектирования и реализации подсистем и отдельных логических АРМов, входящих в состав ИС;
- разработка требований к средствам базового технического обеспечения ИС;
- разработка требований к средствам базового программного обеспечения ИС.

Логическая модель, отображающая деятельность системы управления предприятием, и информационное пространство, в котором эта деятельность протекает, представляют собой "снимок" положения дел (функциональная структура, роли должностных лиц, взаимодействие подразделений, принятые технологии обработки управленческой информации, автоматизированные и неавтоматизированные процессы и т. д.) на момент обследования. Эта модель позволяет понять, что делает и как функционирует предприятие с позиций системного анализа, и затем сформулировать предложения по улучшению ситуации.

Развитие логической модели предметной области, ее последовательное

превращение в модель целевой ИС, позволит интегрировать перспективные предложения руководства и ведущих сотрудников предприятия, экспертов и системных аналитиков, сформировать видение новой, реорганизованной и автоматизированной деятельности предприятия.

Построенная модель является законченным результатом по следующим причинам:

1. Она включает в себя модель существующей неавтоматизированной технологии, принятой на предприятии. Формальный анализ этой модели позволяет выявить узкие места в управлении предприятием и сформулировать рекомендации по его улучшению (независимо от того, предполагается ли дальнейшая разработка автоматизированной системы или нет).
2. Она независима и отделяема от конкретных разработчиков, не требует сопровождения и может быть безболезненно передана другим лицам. Более того, если по каким-либо причинам предприятие не готово к реализации проекта в данный момент времени, модель может быть "положена на полку" до тех пор, пока в ней не возникнет необходимость.
3. Она позволяет осуществлять эффективное обучение новых работников конкретным направлениям деятельности предприятия, так как соответствующие технологии содержатся в модели.
4. С ее помощью можно осуществлять предварительное моделирование перспективных направлений деятельности предприятия с целью выявления новых потоков данных, взаимодействующих процессов и структурных элементов.
5. Она обеспечивает распространение накопленного опыта на других предприятиях, дает возможность унифицировать административно-управленческую и финансовую деятельность этих предприятий.

Модель является не просто реализацией начальных этапов работы и основанием для формирования технического задания на ее последующие этапы. Она представляет собой самостоятельный результат, имеющий большое практическое значение, так как он позволяет дальнейшее применение CASE-технологий для реального проектирования и разработки ИС.

Современные CASE-пакеты имеют широкие возможности инструментального расширения за счет использования стандартных программных средств, что делает их чрезвычайно удобными при разработке программных и информационных систем (рис. 15 и 16).

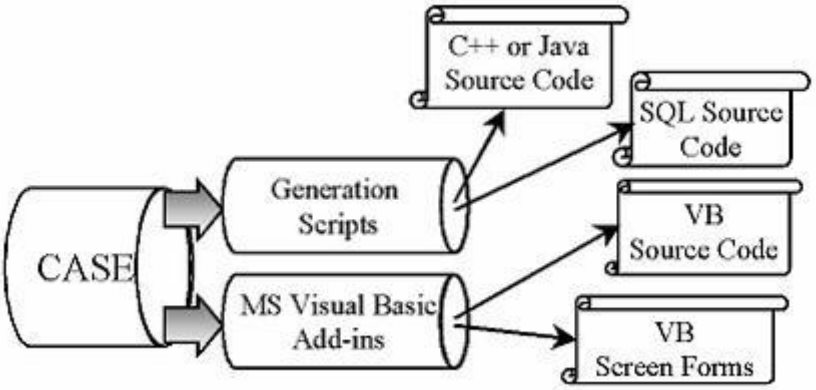


Рисунок 15 – Разработка программных систем

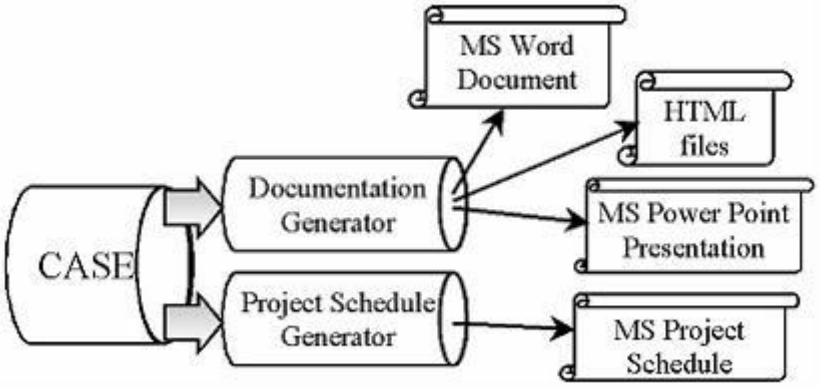


Рисунок 16 – Разработке информационных систем

Для успешного внедрения CASE-средств организация должна обладать нижеследующими качествами:

Культура. Готовность к внедрению новых процессов и взаимоотношений между разработчиками и пользователями, ИТ/ИС-управленцами и пользователями.

Управление. Четкое руководство и организованность по отношению к наиболее важным этапам и процессам внедрения.

Технология. Понимание ограниченности существующих возможностей и способность принять новую технологию.

Если организация не обладает хотя бы одним из перечисленных качеств, то внедрение CASE-средств может закончиться неудачей независимо от степени тщательности следования различным рекомендациям по внедрению.

В качестве примеров популярных CASE-средств - программные средства компании Computer Associates, IBM-Rational Software и Oracle:

- BPwin - моделирование бизнес-процессов;
- ERwin - моделирование баз данных и хранилищ данных;
- ERwin Examiner - проверка структуры СУБД и моделей, созданных в Erwin;
- ModelMart - среда для командной работы проектировщиков;
- Paradigm Plus - моделирование приложений и генерация объектного кода;
- Rational Rose - моделирование бизнес-процессов и компонентов приложений;
- Rational Suite AnalystStudio - пакет для аналитиков данных;
- Oracle Designer (входит в Oracle9i Developer Suite) - высокофункциональное средство проектирования программных систем и баз данных, реализующее технологию CASE и собственную методологию Oracle - CDM. Позволяет команде разработчиков полностью провести проект, начиная от анализа бизнес-процессов через моделирование к генерации кода и получению прототипа, а в дальнейшем и окончательного продукта. Сложное CASE-средство, его имеет смысл использовать при ориентации на линейку продуктов Oracle.

1.8 BPwin - система моделирования бизнес-процессов

Внешние обстоятельства зачастую вынуждают вносить изменения в деятельность организации. Последствия этих изменений должны быть тщательно изучены и осмыслены, прежде чем они станут реальностью. Но процессы, происходящие в современных организациях, нередко оказываются слишком сложными для чисто умозрительного рассмотрения. Построить модель организации и выявить возможные "узкие места" поможет BPwin, мощный инструмент, который используется для анализа, документирования и реорганизации бизнес-процессов.

Мировой лидер в области CASE-технологии предлагает мощное средство системного анализа деловой и производственной активности, позволяющее отслеживать соответствие структуры бизнеса, документооборота, финансовых потоков жестким и динамичным требованиям современной экономики.

Модель, созданная средствами BPwin, позволяет четко документировать различные аспекты деятельности — действия, которые необходимо предпринять, способы их осуществления, требующиеся для этого ресурсы и др. Таким образом, формируется целостная картина деятельности предприятия от моделей организации работы в маленьких отделах до сложных иерархических структур.

Система BPwin поможет повысить конкурентоспособность, оптимизировать процессы управления.

BPwin - это незаменимый инструмент менеджеров и бизнес-аналитиков, а начиная с версии 1.8, в которую включена поддержка диаграмм потоков данных и методики IDEF3 (BPwin Professional), становится в руках системных аналитиков и разработчиков и мощным средством моделирования процессов при создании корпоративных информационных систем.

BPwin обладает интуитивно-понятным графическим интерфейсом, помогает быстро создавать и анализировать модели с целью оптимизации деловых и производственных процессов. Применение универсального графического языка бизнес-моделирования IDEF0 обеспечивает логическую целостность и полноту описания, необходимую для достижения точных и непротиворечивых результатов. Посредством набора графических инструментов BPwin позволяет легко построить схему процесса, на которой показаны исходные данные, результаты операций, ресурсы, необходимые для их выполнения, управляющие воздействия, взаимные связи между отдельными работами.

BPwin поддерживает ссылочную целостность, не допуская определения некорректных связей и гарантируя непротиворечивость отношений между объектами при моделировании. Встроенный механизм вычисления стоимости позволяет оценивать и анализировать затраты на осуществление различных видов деловой активности Механизм вычисления расходов на основе выполняемых действий (Activity-Based Costing, ABC) - это технология, применяемая для оценки затрат и используемых ресурсов. Она

помогает распознать и выделить наиболее дорогостоящие операции для дальнейшего анализа.

BPwin может генерировать отчеты непосредственно в формате MS Excel и Word для последующей обработки и использования в других приложениях. Связь с ERwin (моделирование данных в стандарте IDEF1X) позволяет сократить время проектирования и разработки сложных информационных систем. Для системных аналитиков тесная интеграция BPwin с инструментом проектирования баз данных открывает уникальные возможности по созданию комплексных систем, в которых ERwin служит для описания информационных объектов системы, в то время как BPwin отражает функциональные особенности предметной области. Связывая сущности и атрибуты модели данных с информацией о выполняемых действиях, Вы можете продолжить анализ процессов на новом уровне с одновременной перекрестной проверкой моделей процессов и данных.

Основные характеристики BPwin:

- поддерживает сразу три стандартные нотации - IDEF0 (функциональное моделирование), DFD (моделирование потоков данных) и IDEF3 (моделирование потоков работ). Эти три основных ракурса позволяют описывать предметную область более комплексно;
- позволяет повысить эффективность бизнеса, оптимизировать любые процедуры в компании;
- полностью поддерживает методы расчета себестоимости по объему хозяйственной деятельности (функционально-стоимостной анализ, ABC);
- не дорог, распространён, по нему много информации и компетентных специалистов;
- лёгок в освоении и применении, есть курсы на русском языке
- позволяет облегчить сертификацию на соответствие стандартам качества ISO9000
- является стандартом де-факто, интегрирован с ERwin (для моделирования БД), Paradigm Plus (для моделирования компонентов ПО) и др
- благодаря вышеупомянутой интеграции и поддержке совместной, командной работы над одними и теми же моделями (с помощью ModelMart), не имеет аналогов для крупных проектов.
- Пример модели, построенной в Bpwin интегрирован со средством имитационного моделирования Arena. Имитационное моделирование - создание компьютерной модели системы (физической, технологической, финансовой и т. п.) и проведение на ней экспериментов с целью наблюдения/предсказания. Реальный эксперимент проводить дороже, а зачастую опасно или невозможно;
- содержит собственный генератор отчётов;
- позволяет эффективно манипулировать моделями - сливать и расщеплять их;
- имеет широкий набор средств документирования моделей, проектов.

AllFusion Process Modeler 7 или как он ранее назывался BPwin - мощный программный продукт с помощью которого, можно проводить

моделирование, анализ, описание и последующую оптимизацию бизнес-процессов. С помощью BPwin можно создавать графические модели бизнес-процессов. Графическое изображение схемы выполнения работ, организации документооборота, обмена различными видами информации позволяет визуализировать существующую модель организации бизнеса. Это дает возможность использовать передовые инженерные технологии для решения задач управления организацией.

С помощью BPwin (AllFusion Process Modeler 7) можно организовать подробное документирование всех важных аспекты бизнес-процессов т.е. необходимых действий, способов их осуществления и контроля за ними, необходимыми для этого ресурсами и впоследствии визуализировать полученную информацию. BPwin позволяет повысить эффективность ИТ-решений в бизнесе, проектировщики и аналитики бизнес-моделей получают возможность найти оптимальное соотношение между бизнес-требованиями, корпоративными инициативами, процессами информационной архитектуры и проектированием приложений. С помощью BPwin можно увидеть полную картину организации деятельности предприятия: от количества работы в небольших подразделениях предприятия до сложных функций организации предприятия.

Использование BPwin (AllFusion Process Modeler 7) эффективно использовать в проектах, в которых нужно сделать описание существующих баз предприятия, внедрить на предприятии корпоративные информационные систем и для проведения реорганизации существующих бизнес-проектов. С помощью BPwin можно провести оптимизацию деятельности предприятия и осуществить проверку на соответствие ее стандартам ISO 9000, создать проект организационной структуры, исключить ненужные операции, уменьшить размер издержек и увеличить эффективность. В основе программного продукта BPwin (AllFusion Process Modeler 7) заложены общепринятые технологии моделирования, такие как idef0. Моделирование с помощью методологии idef0 рекомендовано к использованию Госстандартом Российской Федерации и является общепринятым стандартом в США. Наглядность и простота моделей Process Modeler делает значительно более простым взаимодействие между различными участниками бизнес-процессов. Популярность BPwin (AllFusion Process Modeler 7) дает возможность согласовывать функциональные модели в электронном виде. BPwin (AllFusion Process Modeler 7) - это продукт компании Computer Associates, он вместе с ERwin Data Modeler (ERwin), Model Manager (ModelMart) и Data Model Validator (ERwin Examiner), входит в пакет программ AllFusion Modeling Suite. Использование этого программного комплекса позволяет эффективно обеспечить все аспекты моделирования информационных систем.

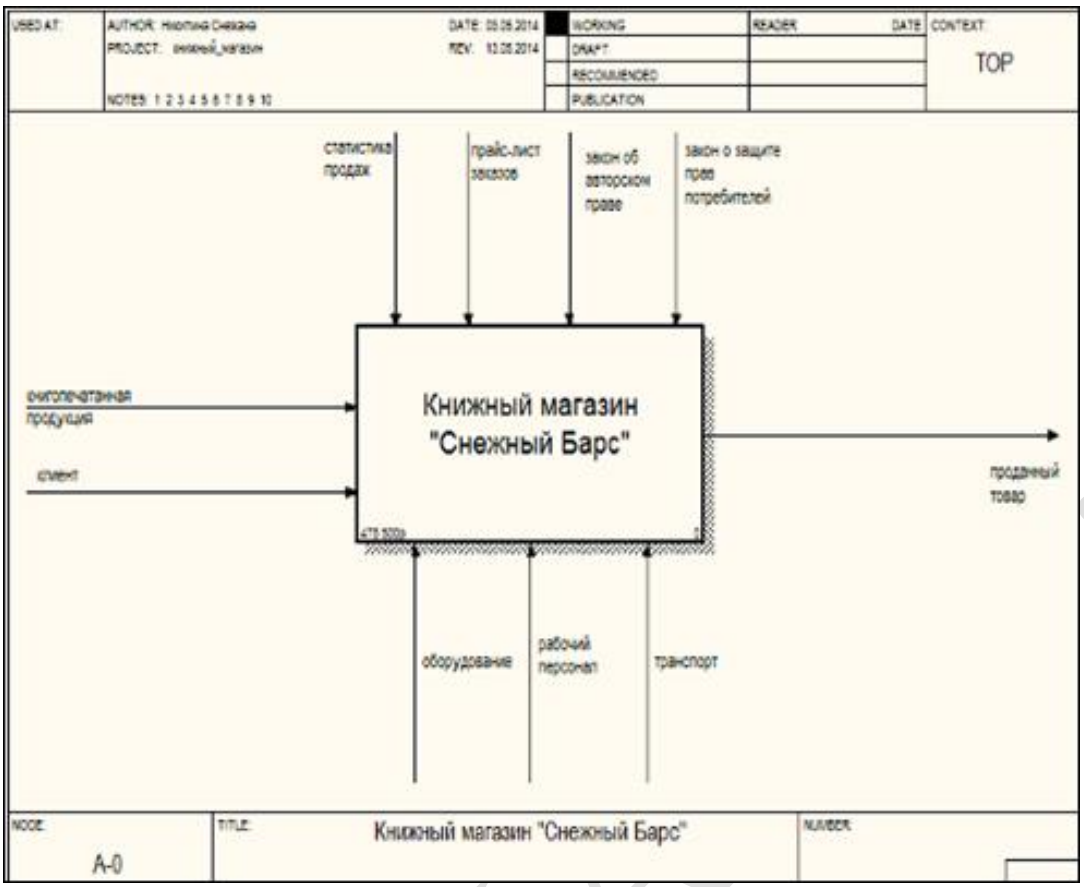


Рисунок 17 – Пример контекстной диаграммы в BPwin

Таким образом, современные CASE-средства вместе с системным программным обеспечением и техническими средствами поддержки образуют *полную среду разработки* информационных систем.

1.9 Внедрение информационных систем

Внедрение корпоративной ИС, разработанной самостоятельно или приобретенной у поставщика, зачастую сопровождается ломкой (перепроектированием) существующих на предприятии бизнес-процессов. Приходится перестраивать их под требования стандартов и логику внедряемой системы. Отметим сразу, что внедрение ИС решает ряд управленческих и технических проблем, однако порождает проблемы, связанные с человеческим фактором.

Внедрение информационной системы, как правило, значительно облегчает управление деятельностью предприятия, оптимизирует внутренние и внешние потоки информации, ликвидирует узкие места в управлении. Однако после того как система успешно установлена, "обкатана" в работе и показала свою эффективность, у части сотрудников выявляется нежелание использовать ИС в работе. В результате проведенного реинжиниринга становится ясно, что некоторые сотрудники в большой степени дублируют работу других или вовсе не нужны. Кроме того, внедрение КИС сопровождается обязательным обучением, но, как показывает российский опыт, желающих переучиваться не так много. Ломка старых навыков и прививание новых - долгий и трудный процесс!

Надо четко понимать, что корпоративная ИС призвана упростить управление организацией, улучшить процессы, усилить контроль и обеспечить этим конкурентные выгоды. Только с такой точки зрения можно оценивать пользу от ее внедрения.

Следуя этой логике, становится понятно, что хотя корпоративная ИС предназначена в целом для обеспечения всех пользователей необходимой информацией, управление разработкой и внедрение КИС является прерогативой высшего руководства компании! Понимают ли это руководители?

Здесь тоже приходится бороться с живучими стереотипами. "Зачем мне корпоративная система, если дела на предприятии и так идут хорошо?". "Зачем, что-то ломать, если все работает?". Но ведь ломать-то чаще всего и не надо. На первом этапе нужно лишь грамотно и корректно формализовать и перенести идентифицированные процессы, в рамках которых живет предприятие, в корпоративную ИС. Подобная формализация лишь отточит, отшлифует удачные маркетинговые и производственные находки, оптимизирует процесс управления и контроля и позволит в дальнейшем проводить целенаправленные изменения.

Внедрение новой ИС - сложный процесс, длящийся от нескольких месяцев для небольших ИС до нескольких лет для ИС больших распределенных компаний с широкой номенклатурой продуктов и большим количеством поставщиков. Успех проекта по разработке (или приобретению) и внедрению ИС во многом зависит от готовности предприятия к ведению проекта, личной заинтересованности и воли руководства, реальной программы действий, наличия ресурсов, обученного персонала, способности к преодолению сопротивления на всех уровнях сложившейся организации.

К настоящему времени сложился стандартный набор приемов внедрения ИС. Основное правило: *выполнять обязательные фазы последовательно и не пропускать ни одной из них*.

Критически важными для внедрения являются следующие *факторы*:

- наличие четко сформулированных целей проекта и требований к ИС;
- наличие стратегии внедрения и использования ИС;
- проведение предпроектного обследования предприятия и построения моделей "Как есть" и "Как будет";
- планирование работ, ресурсов и контроль выполнения плана внедрения;
- участие высшего руководства во внедрении системы;
- проведение работ по внедрению ИС специалистами по интегрированию систем совместно со специалистами предприятия;
- регулярный мониторинг качества выполняемых работ;
- быстрое получение положительных результатов хотя бы в части внедренных модулей ИС или в процессе ее опытной эксплуатации.

Перед началом разработки *проекта внедрения* необходимо:

- максимально формализовать цели проекта внедрения ИС;
- оценить минимально необходимые затраты и статьи расхода;
- установить высокий приоритет проекта внедрения перед остальными текущими проектами;
- наделить руководителя проекта максимально возможными полномочиями;
- провести массовую просветительскую работу с персоналом предприятия с целью довести до каждого важность и необходимость предстоящих преобразований;
- разработать организационные меры для применения новых информационных технологий;
- распределить персональную ответственность по всем этапам внедрения и опытной эксплуатации.

Необходимо также определить функциональные сферы внедрения модулей информационной системы:

- организационное управление;
- организационно-административное обеспечение;
- управление бизнес-процессами;
- управленческий, планово-финансовый и бухгалтерский учет;
- управление персоналом;
- управление документацией;
- управление материально-техническим обеспечением;
- управление связями с клиентами и внешней средой.

Кроме того, что перечислено выше, надо задать *технологические требования* к внедрению ИС:

- *системная платформа* - внедрение и адаптация готового решения от производителя или разработка на заказ в соответствии с техническим заданием заказчика;
- *интегрируемость* - данные хранятся и обрабатываются в едином

информационном пространстве; это обеспечивает их полноту, непротиворечивость, достоверность и возможность многократного использования; система может включать в себя вновь разработанные и уже используемые технологии и приложения;

- *адаптируемость* - система настраивается в соответствии с требованиями заказчика и на особенности информационного поля заказчика;
- *распределенность* - система может эффективно функционировать в территориально удаленных подразделениях и филиалах предприятия;
- *масштабируемость* - система может выполняться в виде каркаса, содержащего базовые модули, и дополняться в соответствии с требованиями изменяющейся внешней и внутренней среды.

Основные фазы внедрения информационной системы:

*Фаза "*Предварительные работы по подготовке проекта внедрения ИС*"*. В ходе предпроектного обследования предприятия (рис. 6.4) происходит сбор подробной информации о структурном построении организации, функциональных связях, системе управления, об основных бизнес-процессах, о потоках внутри предприятия (Control Flow, Doc Flow, Data Flow, Work Flow, Cash Flow), необходимой для построения соответствующих моделей и выбора объектов для автоматизации. Оцениваются сроки, ресурсы, виды и объемы работ, номенклатура и стоимость программно-аппаратных и телекоммуникационных средств, стоимость обучения персонала и т. д.

*Фаза "*Подготовка проекта*"*. После завершения первой фазы осуществляется предварительное планирование и формирование процедур запуска проекта:

- формирование проектной и экспертной групп;
- распределение полномочий и ответственности;
- определение организационно-технических требований к процессу внедрения;
- уточнение спецификаций и ожиданий заказчика;
- обучение группы внедрения, состоящей из специалистов предприятия-заказчика.

Последний, очень важный момент почему-то часто пропускается при составлении плана внедрения. А ведь от него в огромной степени зависит успех всего проекта! После начала финансирования проект считается запущенным к исполнению.

*Фаза "*Концептуальная проработка проекта*"*. В течение этой фазы:

- формируется и утверждается концептуальный проект;
- достигается обязательное однозначное понимание намерений всех участников проекта относительно внедряемой ИС;
- уточняются и конкретизируются цели и задачи проекта;
- определяются размеры прототипа системы;
- согласуются укрупненный план работы, последовательность этапов и условия опытной эксплуатации, планово-финансовые и отчетные показатели.

При этом все указанные действия в обязательном порядке документируются, согласуются и утверждаются всеми заинтересованными и ответственными сторонами.

*Фаза "*Реализация проекта*"*. Во время проведения основных работ по внедрению создается, устанавливается и конфигурируется системная среда, определяются процедуры системного администрирования, устанавливаются основные программно-аппаратные комплексы и приложения. В системе настраиваются организационно-штатные и организационно-функциональные структуры предприятия с использованием таких организационных единиц, как филиал, департамент, отдел, рабочая группа и т.д.

Рисунок 18 - Примерное содержание репозитория проекта внедрения

Осуществляется установка, конфигурирование и настройка сетевых и телекоммуникационных средств, производится перенос данных из прежних локальных систем и формирование интерфейсов с унаследованными и внешними системами. При этом все создаваемые модели, планы, рабочие программные продукты, документация помещаются в сквозной репозиторий (хранилище) проекта внедрения (рис. 18). Важной частью этого репозитория является система документации, формируемая в рамках проекта (рис. 19).

Рисунок 19 - Примерный состав документации по процессу внедрения информационной системы

Отрабатываются системные вопросы безопасности работы системы в многопользовательском режиме. Создаются приложения, шаблоны, отчеты, клиентские формы доступа, распределяются полномочия пользователей. Проводится "прогонка" всех систем в "боевом режиме" с участием всех заинтересованных сторон.

После окончания фазы реализации проект внедрения считается законченным. Информационная система передается в эксплуатацию.

Контрольные вопросы:

1. Что такое "открытая информационная система"?
2. Перечислите основные свойства открытых систем.
3. Охарактеризуйте суть современного процессного подхода к управлению деятельностью предприятия и использования этого подхода при разработке ИС.
4. Что включает в себя понятие "Реинжиниринг бизнес-процессов"?
5. Какие модели и каким образом используются при проектировании информационных систем?
6. Какие программные средства используются для моделирования процессов при разработке информационных систем?
7. На основании каких данных и информации разрабатываются модели состояния AS IS и AS TO BE?
8. Кто в компании занимается вопросами разработки, внедрения и развития ИС? Кто участвует в подготовке технического задания на разработку ИС?
9. Назовите основные этапы проектирования информационных технологий.
10. Перечислите этапы жизненного цикла информационной системы.
11. На каком этапе разработки и внедрения ИС производится обучение персонала компании?
12. Перечислите основные фазы внедрения ИС.

1.10 ГОСТ Р ИСО/МЭК 12207. Основные процессы и взаимосвязь между документами

1.10.1 Методологическая основа ГОСТ Р ИСО/МЭК 12207

Методологическая основа ГОСТ Р ИСО/МЭК 12207 - разбиение процессов на группы, которых в стандарте вводится три.

1. Основные[1]. Это процессы, непосредственно относящиеся к жизненному циклу информационной системы. Можно считать, что это производственные процессы организации.
2. Вспомогательные. Это процессы, предназначенные для поддержки основных процессов. Сами по себе эти процессы организации не нужны - только в связи с основными процессами, которые они обслуживают. Несколько процессов из этой группы связано с управлением качеством.

3. Организационные. Это общекорпоративные процессы, такие как "Обучение" или "Управление". Эти процессы существуют в организации независимо от того, как организовано производство и как устроены вспомогательные процессы.

Таблица из стандарта, показывающая структуру процессов организации, приведена ниже.

Номера процессов на рисунке 20 соответствуют номерам разделов документа.

Идея разделения процессов на группы, как мне кажется, следующая. Деятельность по управлению жизненным циклом информационной системы должна быть интегрирована в существующую практику управления. С этой целью, когда в ходе создания информационной системы возникают смежные задачи, не имеющие выраженной ИТ-специфики, для их решения используются вспомогательные процессы, которые в некоторых случаях должны соответствовать другим, более общим стандартам. В частности, процессы обеспечения качества описаны так, чтобы соответствовать стандартам управления качеством ISO 9001.

Рисунок 20 – Структура процессов жизненного цикла программных систем по ГОСТ Р ИСО/МЭК 12207

Далее, если на протяжении жизненного цикла программной системы возникает задача, которая имеет общекорпоративный характер, например задача обучения, используется соответствующий процесс из стандарта, а не общекорпоративный процесс. Возможно, это решение было принято просто для того, чтобы формально обеспечить замкнутость и, следовательно, самодостаточность стандарта. Как следствие, организационные процессы должны быть как-то увязаны с корпоративными стандартами организации.

Процессы, согласно стандарту, состоят из работ, работы - из задач. Последовательность работ и задач, приведенная в стандарте, не является

жесткой. Необходимо только выдерживать логические связи между работами и задачами.

Цель стандарта - определить полную совокупность процессов, которые могут выполняться в ходе проекта по созданию программной системы. Но поскольку проекты могут сильно различаться, например, по масштабам, сложности, рискам и т. п., допускается для каждого проекта локально видоизменять использующиеся в нем процессы, исключая или добавляя отдельные работы и задачи. Такая деятельность называется в стандарте адаптацией. Стандарт содержит описание процесса собственной адаптации.

1.10.2 Вспомогательные процессы

Среди вспомогательных выделены *пять процессов*, специально разработанных для управления качеством. Каждый из них реализует конкретный механизм обеспечения качества. Вот эти процессы.

Процесс обеспечения качества (6.3)

Этот процесс реализует общие принципы управления качеством. Общие принципы состоят в том, что, во-первых, должен быть разработан и выполнен план работ и задач процесса обеспечения качества. Во-вторых, должны быть выполнены работы по обеспечению качества продукта (т. е. программной системы), всех задействованных процессов и системы качества (если это предусмотрено договором). Перечисленные в стандарте работы по обеспечению качества продукта и процессов направлены на то, чтобы обеспечить соответствие продукта и процессов требованиям, сформулированным в договоре, установленных стандартах и процедурах. Сюда же входят и требования к квалификации персонала.

Процесс верификации (6.4)

Этот процесс обеспечивает, как говорится в стандарте, то, "что программные продукты функционируют в полном соответствии с требованиями или условиями, реализованными в предшествующих работах". Он объединяет все работы по анализу разных объектов, возникающих в основных процессах в ходе конкретного проекта, как то: требований к создаваемой системе, проекту, договору, собственно программе, сборке. В число таких объектов входят и процессы, выбранные для реализации проекта в результате адаптации. Для каждого из перечисленных объектов стандарт приводит набор критериев для анализа. Например, для процессов таких критериев четыре:

a) соответствие и своевременность установления проектных требований к планированию;

b) пригодность, реализуемость, выполнимость в соответствии с планом и условиями договора выбранных для проекта процессов;

c) применимость стандартов, процедур и условий к процессам проектирования;

d) укомплектованность и обученность персонала в соответствии с условиями договора".

Эти работы объединяет возможность привлечения для их выполнения независимой стороны, которая сможет обеспечить незаинтересованную оценку.

Процесс аттестации (6.5)

Процесс аттестации - это "процесс определения полноты соответствия установленных требований (*к процедуре испытаний или тестированию системы. - АБ*), созданной системы или программного продукта их функциональному назначению". Для выполнения этого процесса можно привлекать независимого исполнителя, который подготовит контрольные примеры, тестовые данные, специально отобранных пользователей для испытаний системы. Подчеркнем разницу между верификацией программы и аттестацией системы. Верификация обеспечивает соответствие программы технологиям и стандартам программирования, условиям договора, требованиям устойчивости к ошибкам и т. п. Аттестация же регламентирует деятельность по тестированию программного продукта.

Процесс совместного анализа (6.6)

Как и процесс аудита, процесс совместного анализа включает две стороны, участвующие в договоре: анализирующую и анализируемую. Анализируются управление проектом (состояние проекта, предложения по возможным изменениям в проекте, предложения по переоценке критических ситуаций и т. п.) и технические объекты, т. е. создаваемые программные продукты и услуги. Приведено шесть характеристик программных продуктов, которые необходимо анализировать (законченность, соответствие стандартам, соответствие состояния графику и т. п.). Анализ выполняется периодически в сроки, установленные проектным планом.

Процесс аудита (6.7)

Процесс аудита "является процессом определения соответствия требованиям, планам и условиям договора".

Процесс может "выполняться двумя любыми сторонами, участвующими в договоре, когда одна сторона (ревизующая) проверяет другую сторону (ревизуемую)".

Аудиторские проверки проводятся в сроки, установленные планом проекта. Критерии завершения, взаимные обязательства и результаты согласуются сторонами. Основные цели аудиторской проверки следующие:

- соответствие программных продуктов проектной документации;
- соответствие тестовых данных установленным техническим требованиям;
- завершенность процедуры тестирования и отсутствие замеченных , но не устраненных ошибок;
- соответствие документации стандартам.

Приложение А «Процесс адаптации». Процессом адаптации называется "процесс применения положений настоящего стандарта к условиям реализации конкретного программного проекта". Описан этот процесс в том же стиле, что и остальные процессы. *Адаптация* - обязательная деятельность в ходе применения стандарта на практике. Наличие процесса адаптации подразумевает, что ГОСТ Р ИСО/МЭК 12207

сначала внедряется в организации в целом, а затем для каждого проекта из него "выкраивается" подмножество необходимых процессов.

1.11 Виды внедрения, план внедрения. Стратегии, цели и сценарии внедрения

Многие инновации в бизнесе реализуются посредством проектов с участием IT. При этом не важно, речь идет о незначительных операционных улучшениях или глобальных событиях, связанных с преобразованием целого бизнеса, - в конечном итоге это приводит к *изменению*. В частности, использование новой или измененной услуги также является изменением для бизнеса. Этап Внедрения гарантирует то, что запланированные и спроектированные на предыдущих стадиях жизненного цикла услуги смогут достичь ожидаемых бизнесом и IT результатов на практике. Следует отметить, что в некоторой русскоязычной литературе этап Внедрения называется Преобразованием услуг, так как transition с английского переводится как перемещение/переход. Таким образом, этап Внедрения является чем-то вроде буфера для проверки услуг перед непосредственной эксплуатацией.

Прежде чем говорить о Внедрении, необходимо определить основные термины в его контексте.

Преобразование (Transition) - изменение в состоянии, соответствующее перемещению услуги или конфигурационной единицы из одной стадии жизненного цикла к следующей стадии.

Релиз (Release) - набор аппаратного обеспечения, программного обеспечения, документации, процессов или других компонентов, которые необходимы для внедрения одного или нескольких согласованных изменений в услугах. Содержание каждого релиза управляется, тестируется и развертывается как отдельная сущность.

Запрос на изменение (Request for Change или RFC) - формальное предложение на реализацию Изменения. RFC включает в себя детальное описание предложенного изменения, и может быть записано в бумажном или электронном формате.

Тестирование (Test) - деятельность, которая верифицирует, что конфигурационная единица, услуга, процесс, и т.п., соответствует спецификации или согласованным требованиям.

Сборка(Build) - деятельность по компоновке одной и более конфигурационных единиц для формирования части услуги. Термин Сборка также используется в отношении релиза, который утвержден для распространения. Например, Сборка сервера или Сборка ноутбука.

Развертывание (Deployment) - деятельность, отвечающая за перемещение нового или измененного оборудования, ПО, документации, процесса, и т.п., в среду промышленной эксплуатации.

Поддержка в начале эксплуатации (Early Life Support) - поддержка, предоставляемая в отношении новой или измененной услуги в течение некоторого времени непосредственно после того, как услуга была введена в эксплуатацию. Во время Поддержки в начале эксплуатации поставщик услуг может пересматривать KPI, уровни услуги и наблюдаемые пороговые значения, а также задействовать дополнительные ресурсы для Управления

инцидентами и Управления проблемами.

Среда (Environment) - подмножество ИТ-инфраструктуры, которое используется в различных целях. Для сложных Сред есть возможность совместно использовать конфигурационные единицы, например Среды тестовой и промышленной эксплуатации могут использовать различные разделы на одном майнфрейме. Также используется в термине физической среды для обозначения помещения, кондиционирования, системы питания и т.п.

Среда промышленной эксплуатации (Live Environment) - управляемая среда, содержащая конфигурационные единицы в режиме промышленной эксплуатации, используемые для предоставления услуг.

Среда тестирования (Test Environment) - контролируемая среда, используемая для тестирования конфигурационных единиц, сборок, услуг, Процессов и т.п.

Среда сборки (Build Environment) - контролируемая среда, в которой собираются (компонуются) приложения, услуги и другие сборки перед их передачей в Среду тестирования или Среду промышленной эксплуатации.

Приемка(Acceptance) - формальное соглашение, определяющее, что услуга, процесс, План или другой результат завершен, является правильным, надежным и отвечает установленным требованиям. Приемке обычно предшествует оценка или тестирование, приемка часто обязательна для перехода к выполнению следующего этапа проекта или процесса.

1.12 Модель проектной группы. Роли в модели проектной группы

В настоящее время сложность промышленных приложений и систем такова, что процесс их разработки стал практически неуправляемым. Кроме того, их развертывание на сотнях компьютеров, расположенных в разных местах, значительно раздвигает границы процесса разработки.

Один человек не способен создать приложение масштаба предприятия. Ни один разработчик просто не удержит в голове все требования к системе и варианты проекта. Поэтому сегодня разработкой промышленных систем занимаются проектные группы, и все обязанности распределяются среди членов группы.

MSF (модель проектной группы) — не готовое решение, а каркас, который можно адаптировать для нужд любой организации. Один из элементов этого каркаса - *модель проектной группы.* Она описывает структуру группы и принципы, которым надо следовать для успешного выполнения проекта.

Хотя модель группы разработчиков весьма конкретна, при знакомстве с MSF ее нужно рассматривать в качестве отправной точки. Разные коллективы реализуют этот каркас по-разному, в зависимости от масштаба проекта, размеров группы и уровня подготовки ее членов.

Чтобы проект считался удачным, следует решить определенные задачи:

– *удовлетворить требования заказчика* — проект должен выполнить требования заказчиков и пользователей, иначе ни о каком успехе не может быть и речи, возможна ситуация, когда бюджет и график соблюдены, но проект провалился, так как не выполнены требования заказчика;

– *соблюсти ограничения* — разработчики проекта должны уложиться в финансовые и временные рамки;

– *выполнить спецификации, основанные на требованиях пользователей* — спецификации — это подробное описание продукта, создаваемое группой для заказчика; они представляют собой соглашение между проектной группой и клиентом и регулируют вопросы, касающиеся приложения, в основе этого требования лежит принцип «сделать все, что обещано»;

– *выпустить продукт только после выявления и устранения всех проблем* — не существует программ без дефектов, однако группа должна найти и устранить их до выпуска продукта в свет, причем устранением ошибки считается не только ее исправление, но и, например, занесение в документацию способа ее обхода; даже такой способ устранения проблем предпочтительнее, чем выпуск приложения, содержащего невыявленные ошибки, которые в любой момент могут преподнести неприятный сюрприз пользователям и разработчикам;

– *повысить эффективность труда пользователей* — новый продукт должен упрощать работу пользователей и делать ее более эффективной. Поэтому приложение, обладающее массой возможностей применять которые сложно или неудобно, считается провальным;

– *гарантировать простоту развертывания и управления* — эффективность развертывания непосредственно влияет на оценку пользователем качества продукта, например, ошибка в программе установки может создать у пользователей впечатление, что и само приложение небезгрешно, от проектной группы требуется не только подготовить продукт к развертыванию и гладко провести его, но и обеспечить пользователей поддержкой, организовав сопровождение приложения.

Для достижения этих целей в модели проектной группы выполняемые задачи распределяются *по шести ролям*: менеджмент продукта, менеджмент программы, разработка, тестирование, обучение пользователей и логистика. Люди, выполняющие конкретную роль, должны рассматривать проект в соответствии со своей ролью и обладать необходимой для этого квалификацией. Все роли важны для успеха проекта в целом, и поэтому равноправны. В этой модели нет руководителя всего проекта — есть группа людей, знающих, что нужно делать и делающих это.

Таблица 1– Цели и роли

Цель	Роль
Удовлетворение требований заказчика	Менеджер продукта
Соблюдение ограничений проекта	Менеджер программы
Соответствие спецификациям	Разработчик
Выпуск только после выявления и устранения проблем	Тестер
	Инструктор

Повышение эффективности труда пользователя Простота развертывания и постоянное сопровождение	Логистик

а) *Менеджер продукта* должен вовремя реагировать на потребности заказчика. Его главная задача — сформировать общее представление о поставленной задаче и о том, как ее решать. Он должен ответить на вопрос «Зачем мы делаем все это?» и убедиться, что все члены группы знают и понимают ответ на него. Основная цель этой роли — удовлетворение требований заказчика. Для этого менеджер продукта выступает представителем заказчика в группе разработчиков и представителем группы у заказчика. (На этом этапе важно понимать разницу между заказчиком и пользователем: заказчик платит за создание продукта, а пользователи с ним работают.) Кроме того, менеджер продукта вместе с менеджером программы должны прийти к компромиссному решению относительно функциональных возможностей продукта, сроков его разработки и финансирования проекта.

Как представитель заказчика, менеджер продукта отвечает за выполнение требований заказчика, создание бизнес-сценариев, формирование общего представления о проекте у группы и клиента, а также проверяет, удовлетворяют ли решения группы потребностям заказчика.

Как представитель группы, менеджер продукта отвечает за взаимодействие с заказчиком и управление его ожиданиями. При этом он проводит брифинги с клиентом и главными менеджерами, устраивает встречи с пользователями и демонстрации продукта.

Менеджер продукта сообщает группе предварительную дату выпуска продукта, устанавливаемую заказчиком. Однако в действительно успешных проектах графики составляются «снизу — вверх». При этом группа сама определяет возможную дату выхода приложения, которую менеджер продукта сообщает клиенту.

б) Задача *менеджера программы* — вести процесс разработки с учетом всех ограничений. Руководитель этого направления должен понимать разницу между понятиями «руководитель» и «начальник». В своей книге «Dynamics of Software Development» бывший директор отдела программного менеджмента Microsoft Джим Маккарти пишет: «Запомните, что ваша цель — повысить авторитет каждого члена группы, а не подавить его своим».

Главная обязанность менеджера программы — выполнить все стадии разработки так, чтобы нужный продукт был выпущен в нужное время. Он координирует деятельность других членов группы. И хотя иногда ему придется подгонять своих сотрудников, он не должен и помышлять о диктаторском стиле управления.

Главный менеджер программы составляет график проекта на основе информации, полученной от остальных членов группы. Он координирует этот график с руководителями всех подгрупп. Буферным временем проекта также управляет менеджер программы. Если отдельные части работы выполняются раньше графика или, наоборот, задерживаются, именно

менеджер программы должен выяснить, как это скажется на проекте, и изменить график.

Для выполнения своих обязанностей менеджер программы должен отлично разбираться в деловой стороне проекта и иметь ясное представление о технологиях, необходимых для его выполнения. Естественно, что от руководителя отдела программного менеджмента требуется коммуникабельность и талант организатора.

Программный менеджер компании, где только начинают применять MSF, должен полностью понимать все модели и процессы, повышающие эффективность труда членов группы. И хотя умение руководить нужно всем ролям, особенно это качество важно для менеджера программы. Его авторитет должен стать непререкаемым еще до начала проекта у всех его участников, включая бизнес-отделы и руководство организации. Как правило, группа менеджмента программы управляет ресурсами, используемыми другими ролями, а также составляет и координирует расписания совещаний.

Так как менеджер программы отвечает за набор функциональных возможностей приложения, одна из его обязанностей — определить их набор, необходимый для выполнения требований заказчика. Критичные для проекта требования выявляет менеджер продукта, но набор реализующих их функций определяет менеджер программы. Кроме того, менеджер программы дает заказчику рекомендации, касающиеся дальнейшего развития продукта. Отдел программного менеджмента отвечает за функциональные возможности, изложенные в спецификациях (здесь описано, что будет создано) и в главном плане проекта (определяет, как это будет сделано). Он также координирует работу над этими документами. Тем не менее каждый участник проекта вносит свой вклад в функциональные спецификации и в план проекта.

Менеджер программы должен полностью положиться на опыт остальных сотрудников и только следите за соблюдением всех условий и ограничений.

Менеджер программы отвечает и за бюджет проекта, объединяя требования к ресурсам всех членов группы в единый план расходов. Естественно, его задача — не только разобраться в этих требованиях, но и контролировать реальные затраты, сравнивая их с запланированными. Кроме того, менеджер программы должен регулярно сообщать о состоянии работы всем основным участникам проекта. Ведь одним из симптомов неудачного продукта является отсутствие информации о состоянии проекта, пока деньги на него не кончились. Важно внимать, что решение о дополнительном финансировании может требовать изменения других сторон проекта, а следовательно, его будет принимать менеджер программы совместно с заказчиком.

в) *Разработчики* знакомят остальных членов группы с применяемыми технологиями и собственно создают продукт. В качестве консультантов они предоставляют исходные данные для проектирования, проводят оценку технологий, а также разрабатывают прототипы и тестовые системы,

необходимые для проверки решений и сокращения рисков на ранних стадиях процесса разработки. Чтобы создать продукт определенного качества, разработчикам не следует замыкаться на создании кода, они должны участвовать и в решении прикладной задачи. Они творят не ради творчества, а для реализации требований заказчика. Часто, чтобы полностью разобраться в проекте, приходится создавать прототипы, а чтобы протестировать новую технологию, — испытательные системы, помогающие принять окончательное решение относительно архитектуры приложения. Этим также занимаются разработчики.

Как программисты разработчики отвечают за низкоуровневое проектирование и оценку затрат на реализацию продукта. В большинстве организаций несколько основных разработчиков занимаются и архитектурой приложения. Как правило, это требуется на ранних стадиях проекта, когда уточняются детали функциональных спецификаций и описывается взаимодействие продукта с внешними системами.

Разработчики сами оценивают сроки своей работы. Такая концепция MSF — создание графиков ответственными за выполнение конкретного участка членами группы — называется *составлением расписания «снизу — вверх».* Она позволяет выпустить нужный продукт в нужное время за счет уточнения графиков и повышения ответственности за выполнение работы в запланированные сроки.

Разработчики отвечают и за техническую реализацию проекта — в основном на фазах создания логической и физической модели. На этих стадиях их задача — определить методы реализации функциональных возможностей и заданной архитектуры, а также оценить сроки выполнения этой работы. Заметим, что разработчики не выбирают функции — они только решают, как их реализовать.

Кроме того, на стадии «Планирование» разработчики решают, какое влияние окажет на проект добавление или удаление некоторых функций. Разработчики не участвуют в заключительной стадии проекта — развертывании продукта, однако они должны тесно сотрудничать с логистиками на стадии установки приложения.

г) Задача *тестеров* — испытание продукта в реальных условиях, дабы определить, что в продукте работает и что не работает, и нарисовать таким образом точный «портрет» приложения. Естественно, для проведения тестов нужно отлично разбираться и в требованиях пользователей, и в том, как их удовлетворить.

Тестеры разрабатывают стратегию, планы, графики и сценарии тестирования, которые позволяют убедиться, что все ошибки выявлены и исправлены до выпуска приложения. Ошибкой называют любую проблему, из-за которой продукт не выполняет свои функции. Ею может оказаться и ошибка в коде, называемая «жучком», и отклонение от спецификаций, заданных менеджером программы, и недоработки в документации, подготовленной группой обучения пользователей.

Хотя проверяют качества продукта только тестеры, за выпуск хорошего продукта отвечают все члены проектной группы.

Прочие обязанности тестеров часто упускают из виду. К ним относятся:

– *уведомление об ошибках и их отслеживание* — тестовая группа отвечает не только за управление изменениями, но и за систему выявления ошибок и информирования о них;

– *сборка продукта* — в группе должен быть человек, ответственный за сборку (компиляцию) продукта, и часто такой «главный сборщик» является тестером, он может использовать только код, хранящийся в системе управления версиями; эту рутинную работе удается автоматизировать с помощью сценариев, однако необходимо проверять правильность сборки;

– *выявление и контроль рисков* — это обязанность всех членов группы, менеджер программы должен разработать метод контроля - например, с помощью электронных таблиц Microsoft Excel; тестеры отвечают за работу программы в реальных условиях, поэтому их задача — представить группе анализ рисков с этой точки зрения.

д) *Инструктор.* Цель группы обучения — повысить эффективность труда пользователей. Поэтому инструкторы «принимают сторону» пользователей подобно тому, как менеджеры продукта представляют интересы заказчика. Однако перед пользователями инструкторы выступают в роли представителей проектной группы.

В этом последнем качестве группа обучения отвечает за выпуск удобного, полезного продукта, которому практически не нужна поддержка. Персонал группы тестирует удобство использования продукт та, выявляет проблемы в этой области и проверяет проект пользовательского интерфейса.

Активно участвуя в создании пользовательского интерфейса, инструкторы сокращают затраты на сопровождение продукта и поддержку пользователей. Часто же бывает, что изучению этих расходов практически не уделяется внимания, хотя все расчеты очень просты; чем легче работать с приложением, тем меньше затраты на поддержку.

е) *Логистик* представляет интересы служб поддержки и сопровождения, справочных служб и других служб канала доставки. Он занимается развертыванием продукта и его сопровождением и контролирует продукт с этой точки зрения в процессе проектирования. Кроме того, его задача — составление графиков развертывания приложения. Логистики, менеджеры продукта и менеджеры программы совместно определяют порядок передачи продукта пользователям и организации, после чего логистики готовят их к развертыванию приложения.

Логистик, участвующий в крупном проекте, должен обладать опытом развертывания крупномасштабных приложений на нескольких сотнях компьютеров. Именно поэтому от него требуется коммуникабельность и хорошая техническая подготовка. Логистик руководит всеми сотрудниками, устанавливающими и настраивающими пользовательские системы. Кроме того, он должен уметь координировать установку программного обеспечения и оценивать ее результаты.

Логистик обязан разбираться в инфраструктуре продукта и требованиях к его сопровождению. Его задача — проверить, чтобы все

серверы развертывания и рабочие станции пользователей удовлетворяли требованиям. Обычно данные вопросы учитываются в планах выпуска, развертывания и сопровождения продукта.

Хотя основная задача логистика заключается в плавном развертывании продукта, обязанность руководителя этого направления — составить план сопровождения и эксплуатации и убедиться, что существующие группы способны с этим справиться. Важно обучить не только пользователей, но и персонал справочной службы. Причем последних надо начать знакомить с продуктом еще на ранних стадиях разработки — скажем, на этапе бета-тестирования.

Перед сдачей приложения в эксплуатацию следует составить документацию, определить требования к резервному копированию данных и разработать план восстановления на случай отказа систем. После развертывания логистики в течение некоторого времени консультируют группу сопровождения. Конечно, сложно предсказать все трудности, которые могут возникнуть в процессе эксплуатации приложения, поэтому во всех планах следует предусматривать и порядок действий в экстренном случае.

Помимо перечисленных выше обязанностей, логистик занимается сопровождением промежуточных версий продукта в процессе разработки. Его знания требуются и при изучении поведения приложения в тестовой среде. Кроме того, помощь логистиков необходима при создании тестовых, сертификационных и производственных систем. Они должны также сопровождать приложение в процессе моделирования эксплуатации на этапе бета-тестирования.

Размер группы логистики определяется графиком развертывания, уровнем автоматизации установки, наличием средств автоматического развертывания приложений и другими характеристиками этого процесса.

1.13 Типовые функции инструментария для автоматизации процесса внедрения информационной системы

Целью автоматизации информационных процессов является повышение производительности и эффективности труда работников, улучшение качества информационной продукции и услуг, повышение сервиса и оперативности обслуживания пользователей.

Автоматизация базируется на использование средств вычислительной техники (СВТ) и необходимого ПО. Она позволяет существенно сократить время обслуживания пользователей, значительно повысить уровень их обслуживания, преобразует и видоизменяет отдельные технологические процессы, а порой — все основные традиционно используемые технологии. Автоматизация, способствуя ликвидации многих рутинных операции, повышая комфортность и одновременно эффективность работы, предоставляя пользователям новые, ранее неведомые, возможности работы с информацией, создаёт и новые проблемы, решение которых может быть

осуществлено лишь на базе использования общенаучных методов н широкого использования *новых информационных технологий* (НИТ).

Основные задачи автоматизации информационных процессов направлены на:

– сокращение трудозатрат при выполнении традиционных информационных процессов н операций;

– устранение рутинных операций:

– ускорение процессов обработки и преобразования информации;

– расширение возможностей осуществления статистического анализа и повышение точности учетно-отчётной информации;

– повышение оперативности и качественного уровня обслуживания пользователей;

– модернизацию или полную замену элементов традиционных технологий;

– расширение возможностей организации и эффективного использования информационных ресурсов за счёт применения НИТ (автоматическая идентификация изданий, настольные издательские системы, сканирование текстов, CD и DVD. системы теледоступа и телекоммуникаций, электронная почта, другие сервисы Интернета. гипертекстовые, полнотекстовые и графические машиночитаемые данные и др.);

– облегчение возможностей широкого обмена информацией, участия в корпоративных и других проектах, способствующих интеграции и т.п.

При создании АИС целесообразно максимально унифицировать организуемые системы (подсистемы) для удобства их распространения, модификации, эксплуатации, а также обучения персонала работе с соответствующим ПО. разработка которого для АИС связана с гремя основными факторами:

1) существующей программной средой, состоянием системных, прикладных программных средств, в том числе СУБД;

2) необходимостью проведения новых разработок (нецелесообразность модернизации старых или адаптации заимствованных систем);

3) наличием квалифицированных разработчиков.

Разработка (проектирование) систем автоматизации информационных процессов состоит из двух системных аспектов: *анализа* и *синтеза*. Первый предполагает выделение процессов, подлежащих автоматизации, их изучение, выявление определенных закономерностей, особенностей и др. Он необходим также для определения целей и задач создаваемой системы. Второй аспект подразумевает организацию внедрения НИТ для осуществления, полученных в результате анализа, технических, технологических и программных решений.

Для успешного проведения проектных работ рекомендуется выявить один или несколько прототипов проектируемого объекта, на их основе разработать некоторое количество возможных вариантов (их количество, как правило, в несколько раз больше числа выявленных прототипов). Например, для определения организационно-управленческой структуры

автоматизируемой организации в качестве прототипа можно использовать её структуру.

Затем из полученных вариантов следует отобрать альтернативные разновидности. С учетом местных условий и локальных ограничений сократить оставшиеся варианты, из которых выбрать наилучшие решения.

Автоматизированные информационные системы

Автоматизированная система, согласно ГОСТу. — система, состоящая из взаимосвязанной совокупности подразделений организации и комплекса средств автоматизации деятельности, реализующая автоматизированные функции по отдельным видам деятельности.

Разновидностью автоматизированных систем, широко используемых в самых различных областях человеческой деятельности, являются информационные системы. Основной целью таких систем является хранение, обеспечение эффективного поиска и передачи информации по соответствующим запросам. Информационные системы определяют как взаимосвязанную совокупность средств, методов и персонала, используемых для хранения, обработки и выдачи информации в интересах достижения поставленной цели.

Автоматизированные информационные системы (АИС) — область информатизации, механизм и технология, эффективное средство обработки, хранения, поиска и представления информации потребителю. На каждой ступени развития общества они отражают присущий ему уровень высоких технологий.

Выделяются четыре типа автоматизированных информационных систем:

1. Охватывающий один процесс (операцию) в одной организации.
2. Объединяющий несколько процессов в одной организации.
3. Обеспечивающий функционирование одного процесса в масштабе нескольких взаимодействующих организаций.
4. Реализующий работу нескольких процессов или систем в масштабе нескольких организаций.

Для работы с АИС создают специальные рабочие места пользователей (в том числе работников), получившие название «автоматизированное рабочее место» (АРМ).

Автоматизированное рабочее место — комплекс средств, различных устройств и мебели, предназначенных для решения различных информационных задач, в т.ч. поиска информации, а также выполнения специалистами производственных заданий в соответствующей предметной области. Как правило. АРМ реализуются с использованием компьютерной техники и телекоммуникаций.

Общими требованиями, предъявляемыми к АРМ. являются: удобство и простота общения с ними, в том числе настройка АРМ под конкретного пользователя и эргономнчность конструкции; оперативность ввода, обработки, размножения и поиска документов; возможность оперативного обмена информацией между персоналом организации, с различными лицами и организациями за её пределами; безопасность для здоровья пользователя.

Широкое применение находят АРМ для: подготовки текстовых и графических документов: обработки данных, в том числе в табличной форме; создания и использования баз данных, проектирования и программирования.

Выделяют АРМ руководителя, секретаря, специалиста, технического и вспомогательного персонала и другие. При этом АРМ используют различные операционные системы и прикладные программные средства, зависящие, главным образом, от функциональных задач и видов работ (административно-организационных, управленческих и технологических, персонально-творческих и технических).

Основные требования к аппаратным и программным средствам, используемым в АРМ, заключаются в обеспечении: технологичности выполняемых процедур. «дружественного» интерфейса и эргономнчности (удобное расположение технических средств и мебели, высокое качество визуальной информации, простота осуществления диалога с подсказками при неправильных действиях пользователя, наличие средств печати и тиражирования документов, возможность ведения архива и др.).

АИС можно представить как комплекс автоматизированных информационных технологии, составляющих информационную систему, предназначенную для информационного обслуживания потребителей. Основные компоненты и технологические процессы АИС изображены на рисунке21.

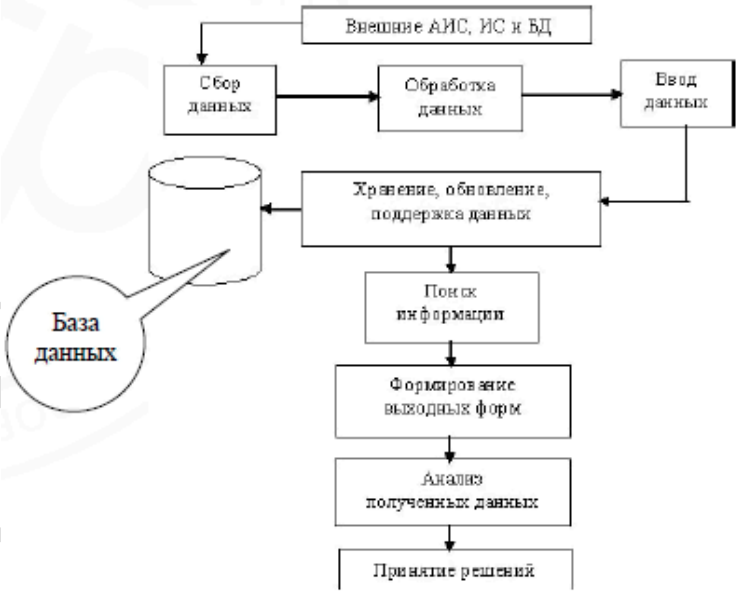


Рисунок 21 – Основные компоненты и технологические процессы АИС

АИС, с точки зрения выполняемых задач и имеющихся возможностей, представляемых пользователям, могут быть как достаточно простыми (элементарные справочные), так и весьма сложными системами (экспертные и др., предоставляющие прогностические решения). Даже простые АИС имеют многозначные структурные отношения между своими модулями, элементами и другими составляющими, что позволяет их отнести к классу сложных систем, состоящих из взаимосвязанных частей (подсистем, элементов), работающих в составе целостной сложной структуры.

Автоматизированная информационная система, ориентированная на персональную информационную поддержку основной деятельности, интегрирующая такие специализированные средства, как поиск, обработка и организация информации, должна строиться с учётом ряда следующих разноплановых особенностей:

1. Используемые ИР ресурсы наряду с оригинальным авторским представлением материала в большинстве своем характеризуются высокой систематизированностью (тематической профильностью источников и ядерностью тематических потоков), а также практически обязательным наличием справочной информации (поисковых образов документов в предметной области — ПОДов. и систем нормативно-справочной информации — рубрикаторов и тезаурусов, обеспечивающих единообразие представления и организации доступа к ИР).

2. Поисковые средства и технологии, используемые для реализации информационных потребностей, определяются типом и состоянием решаемой пользователем задачи основной деятельности: соотношением его знания и незнания об исследуемом объекте. Кроме того, процесс взаимодействия пользователя с системой определяется уровнем знания пользователем содержания ресурса (полноты представления, достоверности источника и т.д.) и функциональных возможностей системы как инструмента. В целом эти факторы обычно сводятся к понятию «профессионализма» — информационного (подготовленный/неподготовленный пользователь) и предметного (профессионал непрофессионал).

3. Вариант схемы автоматизированной информационной системы представлен на рис. 4-2.

4. АИС состоит из технического, программного, информационного, лингвистического и организационного обеспечения (см.пункт 1.1).

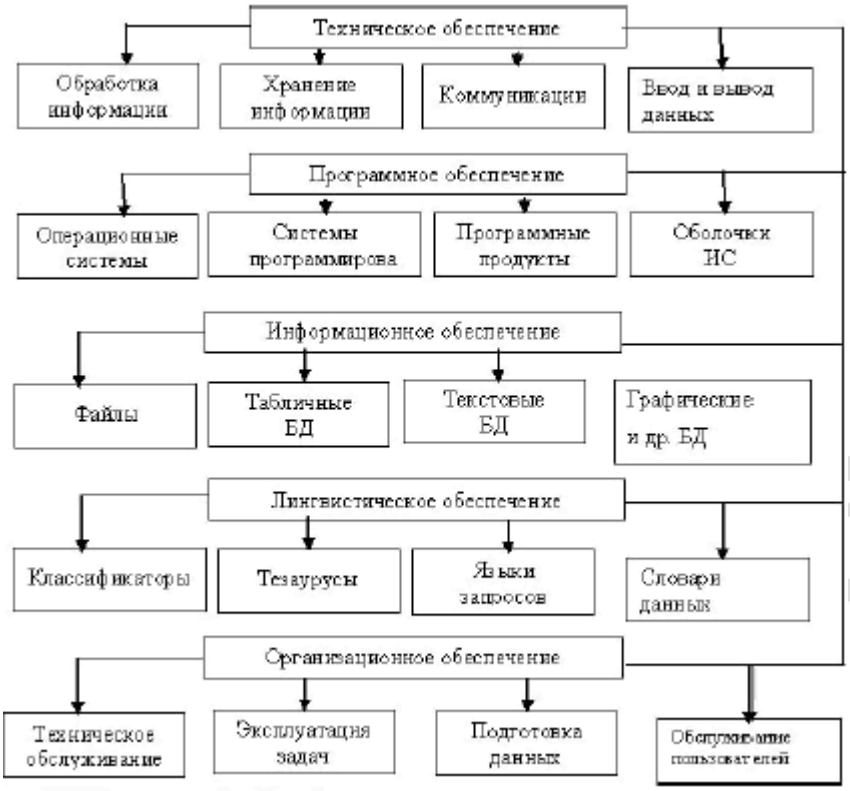


Рисунок 22 – Структура входящих в АИС подсистем

1.14 Оценка качества функционирования информационной системы

Качество ИС связано с дефектами, заложенными на этапе проектирования и проявляющимися в процессе эксплуатации. Свойства ИС, в том числе и дефектологические, могут проявляться лишь во взаимодействии с внешней средой, включающей технические средства, персонал, информационное и программное окружение.

В зависимости от целей исследования и этапов жизненного цикла ИС дефектологические свойства разделяют на *дефектогенность*, *дефектабельность* и *дефектоскопичность*.

Дефектогенность определяется влиянием следующих факторов:

- численностью разработчиков ИС, их профессиональными психофизиологическими характеристиками;
- условиями и организацией процесса разработки ИС;
- характеристиками инструментальных средств и комплексов ИС;
- сложностью задач, решаемых ИС;
- степенью агрессивности внешней среды (потенциальной возможностью внешней среды вносить преднамеренные дефекты, например, воздействие вирусов).

Дефектабельность характеризует наличие дефектов ИС и определяется их количеством и местонахождением. Другими факторами, влияющими на дефектабельность, являются:

- структурно-конструктивные особенности ИС;
- интенсивность и характеристики ошибок, приводящих к дефектам.

Дефектоскопичность характеризует возможность проявления дефектов в виде отказов и сбоев в процессе отладки, испытаний или эксплуатации. На дефектоскопичность влияют:

- количество, типы и характер распределения дефектов;
- устойчивость ИС к проявлению дефектов;
- характеристики средств контроля и диагностики дефектов;
- квалификация обслуживающего персонала.

Стандарты управления качеством промышленной продукции

Международные стандарты серии ISO 9000 разработаны для управления качеством продукции, их дополняют стандарты серии ISO 14000, отражающие экологические требования к производству промышленной продукции. Хотя эти стандарты непосредственно не связаны с CALS-стандартами, их цели - совершенствование промышленного производства, повышение его эффективности - совпадают.

Очевидно, что управление качеством тесно связано с его контролем. Контроль качества традиционно основан на измерении показателей качества продукции на специальных технологических операциях контроля и выбраковке негодных изделий. Однако есть и другой подход к управлению качеством, который основан на контроле качественных показателей не самих изделий, а проектных процедур и технологических процессов, используемых при создании этих изделий.

Такой подход во многих случаях более эффективен. Он требует меньше затрат, поскольку позволяет обойтись без стопроцентного контроля продукции и благодаря предупреждению появления брака снижает производственные издержки. Именно этот подход положен в основу стандартов ISO 9000, принятых ISO в 1987 г. и проходящих корректировку приблизительно каждые пять лет.

Таким образом, методической основой для управления качеством являются международные стандарты серии ISO 9000. Они определяют и регламентируют инвариантные вопросы создания, развития, применения и сертификации систем качества в промышленности. В них устанавливается форма требований к системе качества в целях демонстрации поставщиком своих возможностей и оценки этих возможностей внешними сторонами.

В стандартах ISO 9000 используется определение качества из стандарта ISO 8402: «*Качество* - совокупность характеристик продукта, относящихся к его способности удовлетворять установленные или предполагаемые потребности». Аналогичное определение содержится в ГОСТ 15467-79: «*Качество продукции* - это совокупность свойств продукции, обусловливающих ее пригодность удовлетворять определенные потребности в соответствии с ее назначением». В ISO 9000 вводится понятие *системы качества* (QS - Quality System), под которой понимают документальную систему с руководствами и описаниями процедур достижения качества.

Система качества обычно представляет собой совокупность трех слоев документов:

- описание политики управления для каждого системного элемента;
- описание процедур управления качеством (что, где, кем и когда должно быть сделано);
- тесты, планы, инструкции и т. п.

Сертификация предприятий по стандартам ISO 9001-9003 выполняется некоторой уполномоченной внешней организацией. Наличие сертификата качества - одно из важных условий для успеха коммерческой деятельности предприятий.

Вторичные стандарты включают в себя:

- ISO 9000 - основные понятия, руководство по применению ISO 9001;
- ISO 9004 - элементы систем управления качеством. *Поддерживающие* стандарты предназначены для развития и установки систем качества:
- ISO 10011 - аудит, критерии для аудита систем качества ;
- ISO 10012 - требования для измерительного оборудования;
- ISO 10013 - пособие для развития руководств по управлению качеством.

Часть этих стандартов утверждена как государственные стандарты Российской Федерации. В частности, к ним относятся:

- ГОСТ Р ИСО 9001-96 "Системы качества. Модель обеспечения качества при проектировании, разработке, производстве, монтаже и обслуживании";
- ГОСТ Р ИСО 9002-96 "Системы качества. Модель обеспечения качества при производстве, монтаже и обслуживании";
- ГОСТ Р ИСО 9003-96 "Системы качества. Модель обеспечения качества при окончательном контроле и испытаниях".

В настоящее время разработана новая версия стандартов серии ISO 9000 под названием ISO 9000:2000 Quality management systems (системы управления качеством), в которую включены следующие документы:

- ISO 9000:2000 Fundamentals and vocabulary (основы и терминология);
- ISO 9001:2000 Requirements (требования);
- ISO 9004:2000 Guidelines for performance improvement (руководство по развитию).

Главное отличие новой версии от предыдущей состоит в том, что она обусловлена стремлением упростить практическое использование стандартов, направлена на их лучшую гармонизацию и заключаются в следующем.

В стандарте ISO 9001 минимизируется объем требований к системе качества. Стандарты ISO 9002-9003 из новой версии исключаются. Расширяется круг контролируемых ресурсов, в их число включены такие элементы, как информация, коммуникации, инфраструктура.

Введенные в стандарте ISO 9004 двадцать элементов качества сворачиваются в четыре группы:

- распределение ответственности (management responsibility);
- управление ресурсами (resource management);
- реализация продукции и услуг (product and/or service realization);
- измерения и анализ (measurement, analysis, and improvement).

Сертификация предприятий по стандартам ISO 9001-9003 выполняется некоторой уполномоченной внешней организацией. Наличие сертификата качества - одно из важных условий для успеха коммерческой деятельности предприятий.

Стандарты ISO 14000 являются также системой управления влиянием на окружающую среду; они, как и ISO 9000, реализуются в процессе сертификации предприятий, задают процедуры управления и контроль документации, аудит, подразумевают соответствующее обучение и сбор статистики. Кроме требований заказчиков и покупателей, в них воплощаются внутренние требования организации.

Методы оценки эффективности внедрения CALS-технологий

Чтобы сохранять и наращивать конкурентоспособность предприятия, требуется серьезная автоматизация планирования и управления, а также модернизация проектно-конструкторских и технологических бизнес-процессов на базе информационных технологий.

Информатизация бизнеса - процесс постоянного совершенствования не столько самих информационных систем, сколько управления в целом. Поэтому для оценки инвестиций в автоматизацию компании важно знать

факторы успеха и факторы риска таких проектов, важно соотносить затраты на информационную систему и получаемые преимущества с точки зрения финансовой и организационной перспектив. Уровень таких знаний обеспечит эффективность вложений в информационные технологии.

Современное развитие интеграции производственных данных во всем мире проходит под эгидой *CALS-технологий* - новой концепции развития производственной и коммерческой информатики. В России в последнее время устоялась русскоязычная интерпретация термина CALS - *информационная поддержка изделий* (ИПИ). Ключевым компонентом ИПИ-технологий являются *PDM-технологии* (Product Data Management - управление данными о продукции).

Для всех предприятий сегодня очевидна необходимость перехода на компьютерные технологии поддержки жизненного цикла изделий (ЖЦИ). Однако в настоящее время финансовое положение многих российских предприятий не позволяет применять комплексные решения компьютерного сопровождения этапов проектирования, исследования работоспособности, производства и эксплуатации своих изделий с помощью программного обеспечения ведущих мировых разработчиков CAD/CAE/CAM/ PDM-систем.

В этих условиях внедрение компьютерных технологий целесообразно начинать с решения локальных задач, предполагая их дальнейшее развитие, объединение, комплексирование. И в первую очередь нужно заниматься информатизацией таких задач, решение которых позволит получить в минимальные сроки максимальный экономический эффект.

По мнению отечественных специалистов, применение PDM-систем приводит к существенной экономии и получению дополнительной прибыли, достигаемых за счет сокращения:

- сроков вывода новой продукции на рынок (до 75 %);
- затрат на проектирование сложной продукции (до 30 %);
- доли брака и объема конструктивных изменений (до 70 %);
- расходов на подготовку эксплуатационной и технической документации (до 30-40 %).

Роль PDM в решении комплексных задач автоматизации промышленного предприятия очень велика, поскольку именно на их базе выстраиваются более эффективные бизнес-процессы, обеспечивающие ЖЦИ на различных этапах - маркетинг, проектирование, производство, реализация, эксплуатация; при этом организуется значительно более эффективная работа персонала.

Одна из главных проблем при принятии решения о внедрении PDM-системы заключается в том, что ее внедрение требует значительных материальных затрат, а положительный эффект от ее применения не всегда очевиден. В связи с этим немаловажным является не только оценка результатов применения PDM-технологии на различных предприятиях, но и прогнозирование этих результатов для конкретного предприятия с учетом его специфики.

Существуют три основных варианта (уровня) реализации *системы информационной поддержки ЖЦИ* (СИП ЖЦИ) на предприятиях:

электронный архив технической документации предприятия, система электронного технического документооборота предприятия, система информационной поддержки ЖЦИ.

Внедрение PDM-системы, на базе которой строится СИП ЖЦИ, обычно продолжается несколько лет. Для типового предприятия (3-4 тыс. человек работающих) общая длительность работ по реализации СИП ЖЦИ составляет от 3 до 5 лет. Обычно эти работы делятся на 5-7 этапов. Как правило, используют два подхода к внедрению PDM-системы:

- внедрение PDM-системы во всех подразделениях предприятия с последующим наращиванием ее функциональности;
- полнофункциональное внедрение PDM-системы в отдельных подразделениях предприятия с последующим охватом других подразделений.

Рассмотрим второй способ внедрения PDM-системы. Согласно существующим методикам, в процессе создания СИП ЖЦИ на основе PDM-системы выделяют две фазы:

- первая - предварительное обследование объекта автоматизации (процессов ЖЦ, реализуемых в данной производственной структуре) и разработка архитектуры интегрированной модели изделия и стратегии внедрения ИПИ-технологий;
- вторая - последовательная автоматизация и интеграция отдельных процессов ЖЦИ.

На фазе предварительного обследования объекта автоматизации оценивают текущее состояние предприятия, определяют время, затрачиваемое на разработку документации и ее согласование. При разработке архитектуры интегрированной модели изделия и стратегии внедрения ИПИ-технологий оценивают также предполагаемые затраты на внедрение.

Вторая фаза состоит из последовательных частных проектов автоматизации отдельных этапов ЖЦИ на основе PDM-системы, обеспечивающей управление данными и процессами на этом этапе. При этом PDM-система интегрируется с прикладными системами (например, САПР, АСУП и др.), непосредственно реализующими функции этапа.

Самым первым этапом внедрения PDM-системы является *пилотный проект*. Например, для технологической подготовки производства пилотный проект реализует "полную" автоматизацию этого процесса: обеспечивает перевод в электронный вид всех документов процесса, осуществляет автоматизацию выполнения работ через подсистему управления процессами (workflow) и подготовку сотрудников для работы в этой системе.

По итогам пилотного проекта проводится оценка его результатов. Для этого сравнивается состояние до и после автоматизации (например, оцениваются время и затраты на разработку аналогичных процессов). На основании этой оценки делается прогноз о предполагаемых результатах дальнейшего внедрения системы. Проекты, реализуемые на каждом последующем этапе, заканчиваются передачей их в промышленную эксплуатацию, и, если необходимо, выполняется также корректировка архитектуры и стратегии внедрения СИП ЖЦИ.

Методика оценки эффективности внедрения PDM-технологий должна иметь комплексный характер: помимо экономии традиционно выделяемых производственных ресурсов предприятия (сырье, энергия, труд и др.) необходимо оценивать влияние новой организации работы на такие показатели предприятия, как качество продукции, новые методы обслуживания клиентов, и, в конечном счете, конкурентоспособность и общая капитализация предприятия, что в комплексе достаточно трудно оценить единым количественным показателем. Следовательно, комплексный характер методики должен проявляться и при выборе показателей для оценки изменений. Поэтому предлагается комбинированное использование качественных и количественных показателей. Рассчитывая эффект от внедрения PDM-системы, мы определяем эффективность внедрения не только самой системы PDM, но и новых принципов работы предприятия. Первое предполагает автоматизацию, что приводит к экономии ресурсов, а второе - организационную инновацию.

Для оценки экономической эффективности инвестиций в работы по реализации СИП ЖЦИ, как и прочих ИТ, используются следующие группы методов.

Затратные методы:

- оценка единовременных затрат на внедрение и закупку программно-аппаратных комплексов;
- оценка совокупной стоимости владения информационными системами (Total Cost of Ownership, TCO).

Стандартные экономические методы оценки эффекта:

- оценка возврата инвестиций (Return on Investment, ROI);
- NPV - чистая приведенная стоимость проекта;
- отдача активов;
- цена акционера. Рассмотрим кратко каждый из них.

Оценка единовременных затрат на внедрение и закупку программно-аппаратных комплексов. Этот метод может использоваться для минимизации затрат при заранее ожидаемых результатах. Несмотря на все усилия аналитиков, консультантов и специализированных изданий, большинство предпринимателей и управленцев в России до сих пор интересуется только этими затратами. Видимые расходы включают в себя следующие группы затрат:

- капитальные затраты (на аппаратное и программное обеспечение);
- расходы на управление ИПИ-технологиями;
- расходы на техническую поддержку аппаратного и программного обеспечения (ПО);
- расходы на разработку прикладного ПО внутренними силами;
- командировочные расходы;
- расходы на услуги связи;
- другие группы расходов.

Основной методологический подход к оценке эффективности внедрения ИПИ-технологий заключается в статистической оценке результатов выполнения однородных процессов до и после внедрения

системы или ее соответствующего этапа. При этом большое значение имеет выделение рассматриваемого процесса, учет его влияния на общие результаты предприятия, формирование однородной выборки исходных данных. Для каждого этапа жизненного цикла изделия (ЖЦИ) требуется определять свои показатели эффективности.

В качестве основных факторов эффективности автоматизации производственного процесса можно использовать:

- длительность разработки и согласования (проектирования) ТП;
- затраты на разработку и согласование (проектирование) технологических процессов;
- повышение качества изделия.

В результате инвестиций в работы по реализации СИП ЖЦИ, как правило, получают ускорение введения изменений в конструкторскую и технологическую документацию и уменьшение количества ошибок при автоматизации операций преобразования структуры информации. Но оценить количественно такое качественное улучшение в зависимости от характеристик операций информационной интеграции не представляется возможным. Поэтому при исследовании влияния каких-либо характеристик на эффективность производственного процесса учитывают в основном их влияние на трудоемкость и длительность процесса, предполагая, что их дополнительное положительное влияние на качество продукции только увеличит эффект от внедрения этих ИТ и позволит получить большую эффективность автоматизации.

Оценка эффективности внедрения PDM-системы должна включать два этапа: прогнозирование результатов внедрения и оценку эффективности после внедрения PDM-системы. Предварительная оценка эффекта от планируемого внедрения PDM-системы (как и любой автоматизированной системы) является достаточно сложной инженерно-экономической задачей и требует учета многих факторов. После внедрения PDM-си-стемы оценка эффективности ее внедрения проводится путем сравнения показателей бумажного и электронного документооборотов.

Контрольные вопросы:

1. Чем определяется качество ИС?
2. Какие характеристики качества можно определить?
3. Что определяет показатель качества?
4. Охарактеризуйте дефектологические свойства в зависимости от целей исследования и этапов жизненного цикла ИС: дефектогенность, дефектабельность и дефектоскопичность.
5. Как формируется показатель качества?
6. Какие существуют виды метрических шкал для измерения критериев?
7. Что оценивается с помощью функциональных критериев?
8. Для чего предназначены конструктивные критерии?
9. Расскажите о нормативных документах по оценке качества информационных систем.

10. На чем традиционно основан контроль качества?

11. Что является методической основой для управления качеством ИС?

12. Что представляет собой совокупность документов системы качества?

13. Что включают в себя вторичные стандарты системы качества?

14. Для чего предназначены поддерживающие стандарты?

1.15 Организация процесса обновления в информационной системе. Регламенты обновления

Важное значение в современных информационных системах (ИС) имеет эффективное применение политик обновления. Информационные технологии (ИТ) постоянно развиваются, у ИС появляются новые функции. Постоянно обновляется системное и прикладное программное обеспечение (ПО), при этом, как правило, требования новых версий к оборудованию растут. Кроме того, в программном обеспечении обнаруживаются ошибки, недоработки, которые необходимо устранять, причем таким способом, чтобы по возможности не останавливать функционирование всей ИС на время обновления.

Ресурсы ИС целесообразно разделить по критерию обновляемости на *обновляемые* и *необновляемые*. В данном случае под *обновляемостью* следует понимать возможность и целесообразность изменения части ресурса без полной его замены на новый. Например, современные операционные системы (ОС) являются обновляемыми, так как существует возможность и необходимость обновлять отдельные части системы без ее полной замены. Замена же производится по мере необходимости после завершения жизненного цикла текущей версии ОС. В качестве примера необновляемого ресурса можно привести неуправляемый концентратор, который в случае необходимости меняется в сборе – обновление его программных компонентов или какой-то части элементной базы не предусматривается, да и нецелесообразно. Еще один пример: UTP-кабель, изоляция которого со временем пересыхает.

При этом по отношению ко всей ИС в целом обновлением может являться и замена какого-либо ресурса, входящего в состав ИС. Например, замена того же неуправляемого концентратора на новый (например, управляемый), вполне можно считать обновлением ИС. Этот пример иллюстрирует относительность понятия о необновляемости ресурсов.

1.15.1 Терминология политики обновлений

Виды ресурсов ИС. Итак, политика обновлений относится к политикам ИБ и распространяется на ресурсы, которые бывают пяти видов: подаппаратные; аппаратные; программноаппаратные; прикладные; профили системы.

Подаппаратные ресурсы – это ресурсы, не зависящие от ПО, которые работают в инфраструктуре, обеспечиваемой ими. К ним относятся

принтеры, плоттеры, концентраторы, коммутаторы и т. д. Такие ресурсы могут иметь встроенное ПО.

Аппаратные ресурсы – это оборудование, на котором запускается системное и прикладное ПО, т. е. оборудование мобильного, стационарного или встраиваемого компьютера .

Программно-аппаратные ресурсы формируют платформу, т. е. сочетание оборудования и ОС.

Прикладные ресурсы – это прикладное ПО, запускаемое на определенной платформе.

Кроме того, к прикладным ресурсам отнесем данные, которые содержит ИС.

Профили системы представляют из себя инструкции, руководства пользователя, нормативно-правовую документацию, журналы и протоколы учета, различного рода кадровые документы, связанные с деятельностью ИС [2].

Цели обновления ресурсов. Основные цели обновления ресурсов можно разделить на следующие группы:

• моральное устаревание. Будем считать ресурс морально устаревшим в случае, если он перестает удовлетворительно выполнять свои функции с точки зрения пользователей. Обычно в новых версиях добавляются функции, устраняются недочеты предыдущих версий, улучшаются технические характеристики, дизайн, удобство пользования. Моральному устареванию подвержены все виды ресурсов. При этом скорость морального устаревания различного рода ресурсов может быть весьма разной. Медленнее всего устаревают подаппаратные ресурсы. Например, коммутаторы практически не подвержены моральному устареванию (они, как правило, выходят из строя раньше, чем меняются сетевые протоколы). Из составных частей компьютеров медленнее всего морально устаревают блоки питания и корпуса системного блока.

• физический износ. Физическому износу подвержены ресурсы, в составе которых имеются аппаратные компоненты (т. е. подаппаратные, аппаратные и программно-аппаратные ресурсы). Ускоренный физический износ может проявиться в случае заводского брака при производстве оборудования либо в результате воздействия на изделие внешних раздражителей (например, температуры, влаги, давления, ударов и вибрации, пыли и т. п.). Устранение недостатков актуально для всех видов ресурсов и заключается в улучшении пользовательских свойств, технических характеристик, дизайна, устранении ошибок, неточностей, недоработок и поломок, обнаруженных в предыдущих версиях ресурсов.

• устранение недостатков (поломок оборудования, ошибок ПО, неточностей документации, модификация функциональности).

Очевидно, что понятие о физическом износе не применимо к ПО (т. е. прикладным ресурсам), а устранение ошибок вряд ли можно отнести напрямую к подаппаратным и аппаратным ресурсам, не имеющим в своем составе ПО.

Надо отметить, что составные части аппаратных ресурсов имеют разный период использования. Например, жесткий диск компьютера выходит из строя, как правило, гораздо быстрее, чем центральный процессор или корпус системного блока. И все же вечных аппаратных компонентов не существует, поэтому можно с уверенностью заявлять о том, что рано или поздно любая аппаратная часть износится, сломается и потеряет свои свойства.

Способы обновления ресурсов подразделяются на *полные* и *частичные*. При этом имеет смысл упомянуть, что в данном случае понятие о полном обновлении будет очень относительным, так как в рамках ИС возможно говорить лишь о полной замене того или иного компонента общей ИС, что по смыслу в какой-то степени пересекается с понятием о частичном обновлении.

Стратегии обновления оборудования и программного обеспечения. Возможно три основных стратегии обновления оборудования: аварийная, плановая, аварийно-плановая.

• При аварийной стратегии обновления оборудования оно заменяется новым только в случае серьезной поломки или уязвимости, при которой ремонтировать аппаратные компоненты становится нецелесообразным в силу экономических или каких-либо других причин.

Профилактическая замена оборудования на новое не предусматривается. Кроме того, возможна внеплановая замена оборудования, если оно категорически перестает устраивать по производительности или каким-то другим характеристикам.

• При плановой стратегии обновления оборудования каждому аппаратному компоненту устанавливается срок службы, по истечению которого этот компонент обязательно заменяется новым. При поломке или обнаружении уязвимости ранее этого срока возможность замены на полностью новое оборудование не предусматривается, заменяются лишь некоторые элементы либо же неисправный или уязвимый комплект полностью выводится из эксплуатации до наступления срока планового обновления.

• В большинстве случаев используется аварийно-плановая стратегия обновления, которая предусматривает сочетание установки максимальных сроков службы аппаратных компонентов, но при этом не исключает их замены на новые в случае значительных неисправностей, появления серьезных уязвимостей или неожиданного категорического морального устаревания до истечения планового срока эксплуатации.

ПО также подвержено риску поломки и повреждения, поэтому те же три стратегии обновления справедливы и для него. По отношению к ПО выделим *две основных стратегии обновления*: по уязвимости и моральному устареванию.

Обновление по уязвимости производится для устранения ошибок и недоработок, которые могут приводить к сбоям в работе, а также потенциальным уязвимостям системы. Обычно производители ПО выпускают регулярные обновления, установка которых может производиться вручную или автоматически.

Новые версии оборудования и ПО выходят с двумя целями: устранения ошибок и неточностей существующих версий или для изменения самой парадигмы. При этом далеко не всегда целесообразно обновлять аппаратные компоненты, ОС или прикладное ПО до следующей версии непосредственно после ее появления. Производители обычно продолжают техническую поддержку оборудования и выпуск обновлений для предыдущих версий ПО в течение нескольких лет после появления новых версий.

Длительность исправного функционирования некоторых ресурсов зависит от выполнения профилактических работ, направленных на раннюю диагностику возникающих ошибок и износа. Например, профилактическим работам целесообразно подвергать жесткие диски компьютеров и серверов, а также компоненты систем охлаждения.

Также отметим, что в современных условиях полностью заменить компонент оборудования на аналогичный новый оказывается дешевле, чем подвергать ремонту неисправный. Например, раньше профилактике и ремонту подвергались блоки питания компьютеров, но теперь, в большинстве случаев, их просто меняют на новые.

1.15.2 Суть обновления ресурсов

Обновление подаппаратных ресурсов. Как уже упоминалось, подаппаратные ресурсы (принтеры, плоттеры, концентраторы, коммутаторы, источники бесперебойного питания) могут иметь встроенное ПО. При этом к подаппаратным ресурсам относится также, например, UTP-кабель, который не имеет в своем составе никакого ПО.

Обновление подаппаратных ресурсов может включать взаимодействие с компонентами оборудования или ПО данного ресурса.

Среди подаппаратных ресурсов можно выделить как *обновляемые*, так и *необновляемые*.

В качестве примеров частично не обновляемых ресурсов выше приведены неуправляемый концентратор и UTP-кабель (их можно только заменить целиком). Примером обновляемого подаппаратного ресурса является источник бесперебойного питания. Современный источник бесперебойного питания состоит из блока аккумуляторов и управляющей электроники (с ПО). Поскольку гарантийный срок службы современного блока аккумуляторных батарей, как правило, составляет лишь три года, то наиболее частой операцией обновления аппаратной части источников бесперебойного питания является замена аккумуляторов. К источникам бесперебойного питания может быть применено и обновление программной составляющей – системной прошивки.

Теоретически обновление ресурсов подаппаратного уровня не должно существенно отражаться на режиме функционирования ресурсов ИС более высоких уровней, как правило, ОС и прикладному ПО нет совершенно никакой разницы, по какому сетевому кабелю передаются данные, которые они отправляют в сеть, каким роутером осуществляется маршрутизация пакетов и т. д. На практике же из этого правила бывают существенные исключения. Например, существуют «умные» источники бесперебойного

питания, которые поддерживают взаимодействие с ОС оборудования, к ним подключенного, с целью информирования о произошедших событиях, относящихся к режиму энергообеспечения оборудования, на платформе которого функционирует ОС. Для корректного взаимодействия с такими источниками бесперебойного питания в ОС должен быть установлен специальный драйвер. Обновление системной прошивки источника бесперебойного питания может подразумевать и необходимость обновления драйвера, установленного в ОС.

Отметим также, что если рассматривать манипуляции по замене «умного» источника бесперебойного питания относительно функционирования всей ИС, то такая операция вполне может подходить под понятие об обновлении ИС. Тогда получается, что при замене источника бесперебойного питания новым продуктом другого производителя может возникнуть необходимость полной замены драйвера (и, возможно, прикладного ПО для взаимодействия с данным новым ресурсом).

Обновление аппаратных ресурсов. На оборудовании, из которого состоят аппаратные ресурсы ИС, работает системное и прикладное ПО. Кроме того, в составе некоторых составляющих этого оборудования имеются системные прошивки (например, в видеокарте, SCSI-корзине).

Аппаратные ресурсы ИС обновляемы. Обновление может осуществляться на уровне оборудования и на уровне ПО, когда речь идет о замене системной прошивки. Обновлением аппаратных ресурсов будем считать замену каких-либо компонентов, из которых эти ресурсы состоят. Например, замена вентилятора процессора.

Обновление аппаратных ресурсов в большинстве случаев связано с вмешательством в режим работы ресурсов более высоких уровней. Например, замена видеокарты наверняка вызовет необходимость замены драйвера, с помощью которого она взаимодействует с ОС (и даже может потребоваться пересборка ядра ОС). Данная операция может отразиться и на прикладном ПО: современные игровые программы рассчитаны на определенные видеокарты и замена этого устройства может привести к неработоспособности части из них (и наоборот другие игровые программы с новой видеокартой начнут работать корректно и быстро). Кроме прочего, замена видеокарты практически наверняка вызовет необходимость перезагрузки ОС и всего прикладного ПО.

Обновление программно-аппаратных ресурсов. В объем и содержание понятия «платформа» включаются программно-аппаратные ресурсы, состоящие из сочетания оборудования и работающей на нем ОС. В отдельных случаях в понятие о платформе могут входить иные компоненты (СУБД, Java, Eclipse и т. п.).

Программно-аппаратные ресурсы, как правило, являются обновляемыми.

Под обновлением ресурсов программно-аппаратного уровня будем понимать взаимодействие с ОС или системой управления базами данных. Такие обновления производятся чаще всего с целью устранения найденных

ошибок, улучшения пользовательского интерфейса, добавления новых функциональных возможностей.

Можно разделить обновления по критерию целесообразности применения на два типа: *обязательные* (устраняющие ошибки и уязвимости) и *опциональные* (добавляющие или изменяющие функциональные возможности).

Обновление программно-аппаратных ресурсов может вызвать необходимость взаимодействия и с ресурсами более низких уровней. Например, ОС и серверы управления базами данных имеют тенденцию к увеличению требований к минимально необходимым аппаратным ресурсам для собственной корректной работы. Кроме того, взаимодействие с ресурсами по аппаратного уровня производится с помощью драйверов. Производители ОС стремятся включать в свои продукты широкий спектр драйверов для актуальных на момент выхода новой версии ОС аппаратных и подаппаратных ресурсов, но за несколько десятилетий бурного развития индустрии ИТ скопилось такое количество устройств, что обеспечить поддержку их всех в каждой последующей версии ОС становится малореальным (как для производителей ОС, так и для производителей устройств, тем более что многие производители старых устройств уже прекратили свое существование и, следовательно, любую поддержку выпущенной ранее продукции). Поэтому может возникнуть ситуация, когда после обновления ОС необходимо озаботиться закупкой новых аппаратных или подаппаратных ресурсов.

Современные ОС состоят из множества компонентов, каждый из которых имеет определенную версию. Для корректной работы разных прикладных программ могут требоваться разные версии одного и того же компонента ОС. При этом маловероятно, что разные версии одного и того же компонента ОС могут сосуществовать на одном компьютере. Несколько версий прикладных программ могут быть установленными и запускаться в одной системе.

Обновление прикладных ресурсов. Прикладное ПО является, пожалуй, наиболее интенсивно обновляемым ресурсом. Обновления производятся с целью устранения ошибок, допущенных в предыдущих версиях, а также расширения функциональных возможностей (например, улучшения пользовательского интерфейса).

Прикладное ПО можно разделить на три типа:

• автономные программы, не затрагивающие окружение;

• программы, которые пользуются внешними библиотеками (например, Acrobat Reader пользуется шрифтами, XML);

• интегрированные в ОС программы (например, пакет ПО Microsoft Office).

Для разного типа программ должна быть разная политика обновления.

Автономным программам требуется обновление лишь в том случае, если найдена ошибка.

Они вряд ли могут быть источником нарушения безопасности, так как у них нет взаимодействия с сетевыми интерфейсами (хотя и в такие

программы авторы порой умудряются встраивать «закладки»). Достоинство такого рода ПО: если написано грамотно, то оно будет работать в любой программной среде окружения. В качестве примера грамотно написанной автономной программы можно привести FAR, любая версия которого работает в любой версии MS Windows, а также многие программы-архиваторы. Часть автономных программ относится к категории портируемых между программно-аппаратными платформами, когда исходный код разработан таким образом, что может быть собран и скомпилирован для использования под управлением различных ОС на различных видах оборудования.

Программы, использующие для своей работы внешние библиотеки, очевидно становятся зависимыми от этих библиотек и при их изменении могут стать несовместимыми с ними.

На такого рода программы влияют и имеющиеся во внешних библиотеках ошибки, уязвимости, закладки и т. д.

Интегрированные в ОС прикладные программы, будучи зависимыми от ее компонентов, должны обновляться вместе с ОС.

Обновления прикладного ПО могут влиять на необходимость взаимодействия с ресурсами более низких уровней. Например, новой версии прикладной программы может потребоваться новая версия ОС или какого-либо ее компонента, а также установка нового устройства, которое относится к подаппаратному или аппаратному уровню.

Как и ОС, современные прикладные программы могут состоять из множества компонентов, каждый из которых имеет определенную версию. При этом для корректной работы разных модулей прикладных программ могут требоваться разные версии одного и того же компонента другой прикладной программы. Если необходим запуск этих нескольких прикладных программ на одном компьютере, то необходимо, чтобы на одном и том же компьютере функционировали несколько версий одного и того же компонента системы (причем в некоторых случаях эти версии могут быть созданы разными производителями).

Обновление профилей системы. Функционирование любой ИС должно быть обеспечено нормативно-правовой документацией, которая включает описание использованных в ней стандартов и соответствующие правила эксплуатации, функционал, функциональные требования, разграничение зон ответственности администраторов и пользователей ИС.

1.16 Тестирование программного обеспечения в процессе внедрения и эксплуатации

Хорошая стратегия позволяет избежать хаотичных или избыточных тестовых процедур. Своевременный контроль качества всех компонентов системы это результат тщательно спланированных сроков в стратегии тестирования для каждого этапа обеспечения качества ПО. Это помогает оптимизировать затрачиваемые усилия и сокращать внеплановые расходы.

В начальной стадии проекта проводится первичная оценка сроков и разрабатывается план работ с привязкой к этапам разработки и выпуску релизов. Подбирается набор тестов разных видов и их очередность, оценивается трудозатраты для каждого этапа. План включает:

- выбор методологии и инструментов;
- определение целей и длительности каждого этапа;
- состав работ: охват и применяемые типы тестирования;
- планирование трудозатрат и структура проектной группы;
- разработка тестовой документации и создание отчетности.

Этапы полного цикла тестирования:

Полный цикл тестирования обычно совпадает с итерацией разработки или соответствует ее определенной части. Подход к проверке работоспособности программного продукта похож на оценку продукта от конечного пользователя, поэтому стоит привлекать специалиста к работе на самом раннем этапе — в ходе сбора и анализа требований. Идеально когда обсуждение компонентов системы проходит с участием разработчика, пользователя и QA-аналитика (описывает процесс поиска и исправления ошибок в коде программы, а также документирование процесса и причины ошибки).

– определение цели тестирования: выбрать тестируемые фрагменты и сформулировать задачи тестирования;

– верифицирование метода тестирования: настроить среды и инструменты для тестирования и выполнить отдельные тесты;

– подтверждение правильности сборки: провести предварительные тесты на содержание явных ошибок;

– проведение тестировании: разработать необходимые тесты и выполнить их в ручном или автоматическом режиме;

– оценивание результатов: определить критерии завершения и успешности тестирования;

– улучшение наборов тестов: описать и сохранить тесты, настройки среды и инструменты для последующих циклов.

Всего принято выделять 7 этапов тестирования:

Работа с требованиями. Требования к ПО должны быть систематизированы и тщательно документированы. Процедуры обеспечения качества должны вводиться до начала стадии разработки — на этапе сбора и анализа требований к ПО. Мы протестируем требования к ПО на соответствие бизнес-целям, полноту охвата, уместность использования, целостность и непротиворечивость.

Тщательное изучение требований должно:

– выявить противоречия в требованиях;

– помочь определить потенциальные дефекты в функционале.

Разработка стратегии тестирования. Оценка сроков тестирования, выявление среды тестирования, объединение всей информации, полученной при работе с требованиями.

Тест — лид должен:

– резюмировать полученную информацию,

– оценить сроки тестирования,

– разработать стратегию тестирования: определить виды тестирования, которые можно применить к проекту, проанализировать имеющиеся среды и ресурсы, что имеется для проведения тестирования, описать приоритеты для непредвиденных ситуаций, как и где будет вестись тестовая документация;

– определение среды тестирования: какое оборудование необходимо для тестирования,

– составить план, который содержит описание, с чего начинается и чем заканчивается тестирование, и что будет тестироваться.

Создание тестовой документации. Трудоемкость этого процесса зависит от степени детализации, формата и охвата тестовой документации и может варьироваться. Основная цель тестовой документации — сделать объем и ход выполнения задач прозрачным и понятным для заказчика. Необходимо поддерживать документацию в актуальном виде с учетом новых дополнений и изменений, производимых по ходу проекта. Наши специалисты позаботятся о своевременном создании и регулярном обновлении соответствующей документации, фиксируя все шаги проделанной работы.

Тестовая документация может состоять из:

– тестовых сценариев: что и как будет проверяться при регресс-, дымовом и приемочном тестированиях;

– отчетности: результаты тестирования, списка багов и их серьезность;

– методологий тестирования.

Тестирование прототипа. Тестирование прототипа позволяет снизить риски разработки путем раннего выявления несоответствий бизнес-требованиям, «узких мест» в структуре приложения, проблем с удобством использования и дефектов логики функционала приложения еще до начала разработки. Своевременные изменения выполненные на этапе прототипирования, помогают предотвратить дорогостоящие переделки системы на стадиях разработки.

Основное тестирование. В план процедур по обеспечению качества, могут быть включены различные виды тестирования. Для удобства и вовлеченности в процесс, результаты тестирования обрабатываются в виде отчетов о проделанной работе с описанием найденных дефектов.

Стабилизация. Когда продукт или система практически готова к релизу или эксплуатации, проводится стабилизационное тестирование. Этот этап проводится в наиболее приближенных к реальным условиях или в условиях эксплуатации. Некоторые из функциональных возможностей можно проверить только на этой стадии, например, взаимодействие больших баз данных, оплата в режиме реального времени и т.п.

Эксплуатация и поддержка. Все вносимые в программное обеспечение изменения, должны быть тщательно протестированы. ПО должно продолжать выполнять изначально заложенные в нем бизнес-функции и не нарушать работоспособность остальных функций и всей системы в целом.

Изучив все этапы тестирования сложно выделить главный – каждый из них важен по-своему. Тестирование является долгим и кропотливым процессом, результатом которого является выявленная ошибка в системе. Тестировщик должен четко формулировать свою позицию, почему найденная ошибки — действительно дефект, должен прислушиваться после релиза к позиции конечного пользователя и помнить, что ничто в этом мире не идеально.

Задачи полного цикла тестирования:

Планирование тестов:

- определение требований к тестам;
- оценка рисков;
- выбор стратегии тестирования;
- определение ресурсов;
- создание расписания/последовательностей;
- разработка Плана тестирования;

Дизайн тестов:

- анализ объёма работ;
- определение и описание тестовых случаев;
- определение и структурирование тестовых процедур;
- обзор и оценка тестового покрытия;

Разработка тестов:

- программирование тестовых скриптов;
- определение критичной функциональности в Дизайне и Модели реализации;
- создание и подготовка внешних наборов данных;

Проведение тестов:

- выполнение тестовых процедур;
- оценка выполнения тестов;
- восстановление после сбойных тестов;
- проверка результатов;
- исследование неожиданных результатов;
- запись ошибок;

Оценка тестов:

- оценка покрытия тестовыми случаями;
- оценка покрытия кода;
- анализ дефектов;
- определение критериев завершения и успешности тестирования.

Типы тестирования программного обеспечения:

- *Регрессионное тестирование* — проверка работоспособности ПО после внесения в него изменений.
- *Функциональное тестирование* — проверка соответствия продукта функциональным требованиям и спецификациям.
- *Нагрузочное тестирование* — обеспечение надежности работы ПО при заданных условиях. Измерение производительности, определение системной конфигурации.

• *Модульное тестирование ПО* — полный цикл тестирования отдельных компонентов на возможность интеграции и использования в составе более крупных систем. Тестирование модулей исходных кодов на соответствие требованиям к оформлению.

• *Оптимизационное тестирование* — устранение узких мест с помощью улучшения алгоритмов, а также использования верных технологий и решений.

• *Тестирование интерфейса* — тестирование пользовательского интерфейса программного продукта для проверки его функциональных характеристик, удобства использования, понятности и соответствия определенным стилевым решениям.

Анализ исходного кода в соответствии с предъявляемыми на проекте требованиями

• *Анализ документации* — анализ спецификаций на полноту и достоверность, проверка пользовательских инструкций и документации продукта.

• *Общее (финальное) тестирование* — тестирование методом «черного ящика», основанное на проверке функциональности, которую должен иметь тестируемый продукт в соответствии со спецификацией и документацией.

Ручное и автоматизированное тестирование

Тестирование ПО может выполняться различными методами: методом черного или белого ящика, проводиться по заранее подготовленным сценариям или интуитивно.

Тестирование может быть различных уровней: модульное, системное, интеграции. А также может быть направлено на проверку различных аспектов качества: производительности, безопасности и т.д.

Ниже в таблице приводим краткое сравнение обоих подходов:

Характеристика	Ручное тестирование ПО	Автоматизированное тестирование ПО
Надежность	Результаты менее надежны из-за влияния человеческого фактора	Безошибочные результаты из-за отсутствия влияния человеческого фактора
Скорость выполнения	Медленнее, требует больших трудозатрат	Тестирование происходит автоматически, выполняется быстрее
Стоимость	Оплата труда инженеров по тестированию	Инструменты для разработки решения по автоматизации, работа инженеров по автоматизации
Регулярность запуска тест-кейсов	Тест-кейсы запускаются несколько раз без многократных	Тест-кейсы запускаются регулярно на протяжении долгого периода времени

	повторений	

1.17 Эксплуатационная документация

Эксплуатационная документация – это комплект документов, которые поставляются вместе с системой, комплексом, программным продуктом. Она необходима для ознакомления с правилами сборки, использования, обслуживания и ремонта изделия в процессе эксплуатации.

К эксплуатационным относятся следующие документы:

Ведомость эксплуатационных документов (код вида документа - 20) - содержит перечень эксплуатационных документов на программу.

Формуляр (код вида документа - 30) - содержит основные характеристики программы, комплектность и сведения об эксплуатации программы.

Описание применения (код вида документа - 31) - содержит сведения о назначении программы, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств.

Руководство системного программиста (код вида документа - 32) - содержит сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения.

Руководство программиста (код вида документа - 33) - содержит сведения для эксплуатации программы .

Руководство оператора (код вида документа - 34) - содержит сведения для обеспечения процедуры общения оператора с вычислительной системой в процессе выполнения программы .

Описание языка (код вида документа - 35) - содержит описание синтаксиса и семантики языка.

Руководство по техническому обслуживанию (код вида документа - 46) - содержит сведения для применения тестовых и диагностических программ при обслуживании технических средств.

Руководство системного программиста

Руководство системного программиста относится к эксплуатационно-технической документации. Документ предоставляет сведения для проверки, обеспечения функционирования и конфигурирования программы, если данные возможности обусловлены системным кодом ПО.

Во многих компаниях установка и глубокая настройка некоторых программ обычным пользователям запрещена. Данная работа входит в список обязанностей системного программиста, который проверяет логи, обеспечивает автоматическое резервное копирование данных, ведет статистику производительности и устраняет технические неполадки.

Все перечисленные функции отображаются в документе «Руководство системного программиста». Данный документ разрабатывается и оформляется в соответствии с нормативами ГОСТ 19.503-79, а также

сопутствующими ГОСТ 19.101-77 (Виды программных документов) и ГОСТ 19.105-78 (Общие требования к программным документам).

Если руководство разрабатывается на простую монолитную программу, то документ получается относительно необъемным. Однако крупные аппаратно-программные комплексы требуют описания каждого компонента, чтобы обеспечить их интеграцию, а также работоспособность со сторонним ПО.

В руководстве системного программиста излагают следующую информацию:

– сфера использования и задачи ПО;

– принципы действия;

– системные требования;

– описание процедуры установки компонентов;

– инструкция по настройке ПО;

– методы интеграции установленных программ с их копиями и сторонним ПО;

– периодичность и методика контроля работоспособности;

– методы и порядок обслуживания ПО;

– методы решения вспомогательных задач;

– работа в случаях аварийной ситуации.

В зависимости от типа программы и способа ее настройки также может включаться другая информация (интерфейс, утилиты командной строки, язык для скриптов и т.д.).

Разработка руководства программиста

Руководство программиста относится к эксплуатационно-технической документации. Разрабатывается такой документ для программных продуктов. Предназначен для ознакомления программистом, который будет решать те или иные задачи, связанные с эксплуатацией данной программы.

Руководство программиста необходимо в нескольких случаях:

– программа, на которую составляется документация, представляет собой среду разработки или библиотеку.

– данный программный продукт предоставляет платформу для написания типовых программ или систем.

– распространение продукта проходит совместно с программным кодом или же происходит его постоянная модификация разработчиком.

При помощи такого документа программисту должна быть представлена вся необходимая информация, которая может быть использована для создания собственных программных продуктов на базе данной системы. Информация должна быть предоставлена в достаточном количестве. Разработчики же при помощи руководства программиста имеют возможность зафиксировать текущее состояние выпускаемого продукта, чтобы избежать путаницы при выпуске новых продуктов.

К типичным задачам такого документа относится:

– Уточнение и пояснение специалисту текущего состояния объектов, их местонахождения и методов взаимодействия. Также руководство

программиста должно четко разграничить объекты, которые изначально внесены в систему, и объекты, которые программист создает самостоятельно.

– Перечисление дополнительных средств разработки, которые потребуются при работе, кроме текущего продукта.

– Уточнение требований к системе, программной среды, а также средств, необходимых для запуска.

Руководство программиста должно составляться грамотным профессионалом, знакомым со спецификой работы конкретного программного продукта и правилами составления подобных документов, регламентируемых по ГОСТ 19.504-79.

Требования к заполнению руководств программиста установлены соответствующим государственным стандартом. Структура такого документа должна включать в себя:

– Предназначение и условия эксплуатации программного продукта.

– Основные характеристики программы.

– Методы обращения к программному продукту.

– Основная входная и выходная информация.

– Сообщения.

На данный момент руководство программистов включает также различные схемы - схемы баз данных, диаграммы классов, графы вызова функций.

Разработка руководства оператора ГОСТ

Руководство оператора ГОСТ чаще всего рассматривается как документ, в котором указаны конкретные действия оператора. Основная задача оператора – в режиме «on-line» осуществлять обслуживание системы или программного обеспечения входящего в систему, поэтому руководство оператора частично объединяет в себе информацию, предназначенную для пользователя и администратора системы (программы).

Типовая структура руководства оператора регламентируется ГОСТ 19.505-79 и включает:

– Назначение программы

– Условия выполнения программы

– Выполнение программы

– Сообщения оператору

В зависимости от особенностей допускается вводить новые разделы или объединять существующие.

По своему изложению руководство оператора должно содержать минимально необходимую информацию, обеспечивающую нормальный режим работы оператора. В документе должно быть минимальное количество, а в идеале вообще не должно быть разделов содержащих теоретический материал не несущий никакой смысловой нагрузки.

Разработка формуляра по ГОСТ

Формуляр – документ, содержащий основные сведения о системе и отражающий её текущее состояние. Формуляр может разрабатываться как на всю систему в целом, так и на всё её отдельные компоненты. Требования к

наличию формуляра на систему, оборудование или изделие предъявляются стандартами ГОСТ 34.201-89 и РД 50-34.698-90.

ГОСТ РД 50-34.698-90 так же предъявляет требования к содержанию формуляра:

– общие сведения;

– основные характеристики;

– комплектность;

– свидетельство о приемке;

– гарантийные обязательства;

– сведения о состоянии АС;

– сведения о рекламациях.

Формуляр единственный документ, который ведется на всем жизненном цикле системы. Чаще всего ответственным за ведение формуляра является ответственный за эксплуатацию системы (составной части автоматизированной системы). Большинство информации в формуляр заносится «от руки» на всём протяжении функционирования системы (АСУ).

Ведомость эксплуатационных документов

Ведомость эксплуатационных документов (сокращенно ВЭ) – это документ, входящий в пакет эксплуатационной документации и регламентирующий ее комплектность, наименования и правила хранения. Составление данного документа на разработанную автоматизированную систему или программу предусмотрено ГОСТ 34.201-89.

Руководство по техническому обслуживанию

Руководство по техническому обслуживанию относится к эксплуатационной документации, которое содержит описание функций, общие указания к применению и обслуживанию программы.

Главная задача документа — обеспечение персонала информацией о правильном применении и техническом обслуживании программы. Руководство по техническому обслуживанию разрабатывается для пользователей программы, а также специалистов, которые занимаются обслуживанием программного обеспечения. В документе отображаются сведения для использования диагностических, тестовых и других типов программ, которые применяются при техническом обслуживании. Документ разрабатывается в соответствии со стандартами ГОСТ 19.105-78 (общие требования к программным документам) и ГОСТ 19.508-79 (требования к оформлению руководства по техническому обслуживанию).

Руководство по техническому обслуживанию является частью общей документации на программное обеспечение и сопутствующее оборудование, а также входит в состав руководства по эксплуатации.

В зависимости от типа оборудования могут вводиться дополнительные разделы: хранение, транспортировка, способы утилизации и т.д. Использование рисунков, схем и фотографий допускается в рамках требований соответствующих нормативов.

Даже если интерфейс администратора интуитивно понятен, однако руководство по техническому обслуживанию остается обязательным документом. В особенности это касается программ и оборудования, которое

обслуживается несколькими людьми. Для каждой группы следует прописывать уровень доступа, выполняемые функции, требования к обслуживанию и т.

1.18 Качество ПО. Функциональность ПО

а) *Качество ПО*

Качество программного обеспечения - совокупность свойств ПО, обусловливающих его пригодность удовлетворять определенные потребности пользователей и специалистов, участвующих в создании и сопровождении ПО.

Из приведенной формулировки следует, что не все свойства ПО входят в его качество, а только та их совокупность, которая определяется потребностью в этом ПО. Качество программного продукта можно определить как «пригодность к использованию». Качество должно гарантироваться процессом разработки. Контроль качества программного продукта — это систематические действия, подтверждающие пригодность к использованию программного продукта в целом. Цель контроля качества — дать количественные меры качества программной системы.

Под *свойством (характеристикой) ПО* понимается объективная особенность ПО (программ и документации), проявляющуюся при его разработке, эксплуатации и сопровождении. Свойства ПО можно условно разделить на функциональные и конструктивные. Функциональные свойства отражают возможности и специфику применения программы и обусловливают степень ее соответствия своему целевому назначению. Они характеризуют программу с точки зрения того, как в действительности она выполняется. Конструктивные свойства программы более или менее не зависят от ее функциональных возможностей и назначения. Они характеризуют программу с точки зрения того, как в действительности она сконструирована.

Для объективной оценки качества ПО его свойства необходимо охарактеризовать количественно. *Показатель качества ПО* — количественная характеристика свойства ПО, входящая в состав его качества и рассматриваемая применительно к определенным условиям его создания, эксплуатации и сопровождения. Наряду с показателями качества могут использоваться качественные (словесные) оценки, называемые признаками.

Показатели качества по количеству характеризуемых свойств могут быть *единичными* и *комплексными* (групповыми). Единичный показатель относится только к одному из свойств, тогда как комплексный характеризует несколько свойств ПО.

Характеристики качества ПО:

Функциональность (Functionality) - определяется способностью ПО решать задачи, которые соответствуют зафиксированным и предполагаемым потребностям пользователя, при заданных условиях использования ПО. Т.е. эта характеристика отвечает то, что ПО работает исправно и точно,

функционально совместимо соответствует стандартам отрасли и защищено от несанкционированного доступа.

Надежность (Reliability) – способность ПО выполнять требуемые задачи в обозначенных условиях на протяжении заданного промежутка времени или указанное количество операций. Атрибуты данной характеристики – это завершенность и целостность всей системы, способность самостоятельно и корректно восстанавливаться после сбоев в работе, отказоустойчивость.

Удобство использования (Usability) – возможность легкого понимания, изучения, использования и привлекательности ПО для пользователя.

Эффективность (Efficiency) – способность ПО обеспечивать требуемый уровень производительности, в соответствии с выделенными ресурсами, временем и другими обозначенными условиями.

Удобство сопровождения (Maintainability) – легкость, с которой ПО может анализироваться, тестироваться, изменяться для исправления дефектов для реализации новых требований, для облегчения дальнейшего обслуживания и адаптирования к имеющемуся окружению.

Портативность (Portability) – характеризует ПО с точки зрения легкости его переноса из одного окружения (software/ hardware) в другое.

Модель качества программного обеспечения

На данный момент, наиболее распространена и используется многоуровневая модель качества программного обеспечения, представленная в наборе стандартов ***ISO 9126***. На верхнем уровне выделено 6 основных характеристик качества ПО, каждую из которых определяют набором атрибутов, имеющих соответствующие метрики для последующей оценки.

Рисунок 23 – Модель качества программного обеспечения (ISO 9126-1)

б) *Функциональность ПО*

Функциональность ПО – совокупность свойств ПС, определяемая наличием и конкретными особенностями набора функций, способных удовлетворять заданные или подразумеваемые потребности качества наряду с ее надежностью как технической системы.

Завершенность программного средства (ПС) является общим примитивом качества ПС для выражения и функциональности и надежности ПС, причем для функциональности она является единственным примитивом.

Функциональность ПС определяется его *функциональной спецификацией*. Завершенность ПС как примитив его качества является мерой того, в какой степени эта спецификация реализована в разрабатываемом ПС. Обеспечение этого примитива в полном объеме означает реализацию каждой из функций, определенной в функциональной спецификации, со всеми указанными там деталями и особенностями. Все рассмотренные ранее технологические процессы показывают, как это может быть сделано.

Однако в спецификации качества ПС могут быть определены несколько уровней реализации функциональности ПС: может быть определена некоторая упрощенная (начальная или стартовая) версия, которая должна быть реализована в первую очередь; могут быть также определены и несколько промежуточных версий. В этом случае возникает дополнительная

технологическая задача: организация наращивания функциональности ПС. Здесь важно отметить, что разработка упрощенной версии ПС не есть разработка его прототипа. Прототип разрабатывается для того, чтобы лучше понять условия применения будущего ПС, уточнить его внешнее описание. Он рассчитан на избранных пользователей и поэтому может сильно отличаться от требуемого ПС не только выполняемыми функциями, но и особенностями пользовательского интерфейса. Упрощенная же версия разрабатываемого ПС должна быть рассчитана на практически полезное применение любыми пользователями, для которых предназначено это ПС. Поэтому главный принцип обеспечения функциональности такого ПС заключается в том, чтобы с самого начала разрабатывать ПС таким образом, как будто требуется ПС в полном объеме, до тех пор, когда разработчики будут иметь дело непосредственно с теми частями или деталями ПС, реализацию которых можно отложить в соответствии со спецификацией его качества. Тем самым, и внешнее описание и описание архитектуры ПС должно быть разработано в полном объеме. Можно отложить лишь реализацию тех программных подсистем (определенных в архитектуре разрабатываемого ПС), функционирования которых не требуется в начальной версии этого ПС. Реализацию же самих программных подсистем лучше всего производить методом целенаправленной конструктивной реализации, оставляя в начальной версии ПС подходящие имитаторы тех программных модулей, функционирование которых в этой версии не требуется. Допустима также упрощенная реализация некоторых программных модулей, опускающая реализацию некоторых деталей соответствующих функций. Однако такие модули с технологической точки зрения лучше рассматривать как своеобразные их имитаторы (хотя и далеко продвинутые).

Достигнутый при обеспечении функциональности ПС уровень его завершенности на самом деле может быть не таким, как ожидалось, из-за ошибок, оставшихся в этом ПС. Можно лишь говорить, что требуемая завершенность достигнута с некоторой вероятностью, определяемой объемом и качеством проведенного тестирования. Для того чтобы повысить эту вероятность, необходимо продолжить тестирование и отладку ПС. Однако, оценивание такой вероятности является весьма специфической задачей, которая пока еще ждет соответствующих теоретических исследований.

Функциональная спецификация программного средства

С учетом назначения функциональной спецификации ПС и тяжелых последствий неточностей и ошибок в этом документе, функциональная спецификация должна быть математически точной. Это не означает, что она должна быть формализована настолько, что по ней можно было бы автоматически генерировать программы, решающие поставленную задачу. А означает лишь, что она должна базироваться на понятиях, построенных как математические объекты, и утверждениях, однозначно понимаемых разработчиками ПС. Достаточно часто функциональная спецификация формулируется на естественном языке. Тем не менее, использование

математических методов и формализованных языков при разработке функциональной спецификации весьма желательно.

Функциональная спецификация состоит из трех частей:

– описания внешней информационной среды, к которой должны применяться программы разрабатываемой ПС;

– определение функций ПС, определенных на множестве состояний этой информационной среды (такие функции будем называть *внешними функциями* ПС);

– описание нежелательных (исключительных) ситуаций, которые могут возникнуть при выполнении программ ПС, и реакций на эти ситуации, которые должны обеспечить соответствующие программы.

В первой части должны быть определены на концептуальном уровне все используемые каналы ввода и вывода и все информационные объекты, к которым будет применяться разрабатываемое ПС, а также существенные связи между этими информационными объектами. Примером описания информационной среды может быть концептуальная схема базы данных или описание сети датчиков и приборов, которой должна управлять разрабатываемая ПС.

Во второй части вводятся обозначения всех определяемых функций, специфицируются все входные данные и результаты выполнения каждой определяемой функции, включая указание их типов и заданий всех соотношений (или ограничений), которым должны удовлетворять эти данные и результаты. И, наконец, определяется семантика каждой из этих функций, что является наиболее трудной задачей функциональной спецификации ПС. Обычно эта семантика описывается неформально на естественном языке - примерно так, как это делается при описании семантики многих языков программирования.

В третьей части должны быть перечислены все существенные случаи, когда ПС не сможет нормально выполнить ту или иную свою функцию (с точки зрения внешнего наблюдателя). Примером такого случая может служить обнаружение ошибки во время взаимодействия с пользователем, или попытка применить какую-либо функцию к данным, не удовлетворяющим соотношениям, указанным в ее спецификации, или получение результата, нарушающего заданное ограничение. Для каждого такого случая должна быть определена (описана) реакция ПС.

1.19 Метрические характеристики качества разработки программ

Ме́трика програ́ммного обеспе́чения (англ. software metric) — мера, позволяющая получить численное значение некоторого свойства программного обеспечения или его спецификаций.

В настоящее время в программной инженерии еще не сформировалась окончательно система метрик. Действуют разные подходы к определению их набора и методов измерения.

Система измерения включает метрики и модели измерений, которые используются для количественной оценки качества ПО.

При определении требований к ПО задаются соответствующие им внешние характеристики и их атрибуты (подхарактеристики), определяющие разные стороны управления продуктом в заданной среде. Для набора характеристик качества ПО, приведенных в требованиях, определяются соответствующие метрики, модели их оценки и диапазон значений мер для измерения отдельных атрибутов качества.

Согласно стандарту метрики определяются по модели измерения атрибутов ПО на всех этапах ЖЦ (промежуточная, внутренняя метрика) и особенно на этапе тестирования или функционирования (внешние метрики) продукта.

Остановимся на классификации метрик ПО, правилах для проведения метрического анализа и процесса их измерения.

Существует три типа метрик:

- метрики программного продукта, которые используются при измерении его характеристик - свойств;
- метрики процесса, которые используются при измерении свойства процесса ЖЦ создания продукта.
- метрики использования.

Метрики программного продукта включают:

- внешние метрики, обозначающие свойства продукта, видимые пользователю;
- внутренние метрики, обозначающие свойства, видимые только команде разработчиков.

Внешние метрики продукта - это метрики:

- надежности продукта, которые служат для определения числа дефектов;
- функциональности, с помощью которых устанавливаются наличие и правильность реализации функций в продукте;
- сопровождения, с помощью которых измеряются ресурсы продукта (скорость, память, среда);применимости продукта, которые способствуют определению степени доступности для изучения и использования;
- стоимости, которыми определяется стоимость созданного продукта.

Внутренние метрики продукта включают:

- метрики размера, необходимые для измерения продукта с помощью его внутренних характеристик;
- метрики сложности, необходимые для определения сложности продукта;
- метрики стиля, которые служат для определения подходов и технологий создания отдельных компонентов продукта и его документов.

Внешние и внутренние метрики задаются на этапе формирования требований к ПО и являются предметом планирования и управления достижением качества конечного программного продукта.

Стандарт ISO/IEC 9126-2 определяет следующие типы мер:

- мера размера ПО в разных единицах измерения (число функций, строк в программе, размер дисковой памяти и др.);

- мера времени (функционирования системы, выполнения компонента и др.);
- мера усилий (производительность труда, трудоемкость и др.);
- мера учета (количество ошибок, число отказов, ответов системы и др.).

Специальной мерой может служить уровень использования повторных компонентов и измеряется как отношение размера продукта, изготовленного из готовых компонентов, к размеру системы в целом. Данная мера используется также при определении стоимости и качества ПО. Примеры метрик:

- общее число объектов и число повторно используемых;
- общее число операций, повторно используемых и новых операций;
- число классов, наследующих специфические операции;
- число классов, от которых зависит данный класс;
- число пользователей класса или операций и др.

При оценке общего количества некоторых величин часто используются среднестатистические метрики (среднее число операций в классе, наследников класса или операций класса и др.).

Как правило, меры в значительной степени являются субъективными и зависят от знаний экспертов, производящих количественные оценки атрибутов компонентов программного продукта.

Примером широко используемых внешних метрик программ являются *метрики Холстеда* - это характеристики программ, выявляемые на основе статической структуры программы на конкретном языке программирования: число вхождений наиболее часто встречающихся операндов и операторов; длина описания программы как сумма числа вхождений всех операндов и операторов и др.

На основе этих атрибутов можно вычислить время программирования, уровень программы (структурированность и качество) и языка программирования (абстракции средств языка и ориентация на проблему) и др.

В качестве метрик процесса могут быть время разработки, число ошибок, найденных на этапе тестирования и др. Практически используются следующие метрики процесса:

- общее время разработки и отдельно время для каждой стадии;
- время модификации моделей;
- время выполнения работ на процессе;
- число найденных ошибок при инспектировании;
- стоимость проверки качества;
- стоимость процесса разработки.

Метрики использования служат для измерения степени удовлетворения потребностей пользователя при решении его задач. Они помогают оценить не свойства самой программы, а результаты ее эксплуатации - эксплуатационное качество. Примером может служить - точность и полнота реализации задач пользователя, а также затраченные ресурсы (трудозатраты, производительность и др.) на эффективное решение задач

пользователя. *Оценка требований* пользователя проводится с помощью внешних метрик.

1.20 Определение надежности ПО

Впервые дискуссии о надежности программного обеспечения в нашей стране развернулись в начале 80-х годов прошлого века после появления на прилавках книжных магазинов перевода монографии известного специалиста в области разработки ПО Гленфорда Майерса. Нельзя сказать, что до этого момента проблема надежности ПО не осознавалась. Просто в упомянутом труде она была четко сформулирована, показана ее суть и намечены основные пути решения. Через некоторое время в свет вышла книга советского ученого Владимира Васильевича Липаева, посвященная той же теме, а следом за ней - перевод руководства еще одного известного зарубежного специалиста Роберта Гласса. Следует отметить, что несмотря на то, что со времени выхода этих изданий прошло более 30 лет, вопросы, поднятые в них, а также основные определения, принципы и предлагаемые методы актуальны до сих пор.

Современный международный стандарт, принятый в качестве государственного российского нормативного документа в 2015 году (ГОСТ Р ИСО/МЭК 25010-2015) дает следующее определение: «надежность (reliability): Степень выполнения системой, продуктом или компонентом определенных функций при указанных условиях в течение установленного периода времени».

Примечания:

- Это определение было адаптировано из (ИСО/ МЭК/ИИЕЕ 24765).
- В программном обеспечении износа не происходит. Проблемы с надежностью возникают из-за недостатков в требованиях, при разработке и реализации или из-за изменений условий использования.
- Характеристики функциональной надежности программного обеспечения включают в себя готовность и либо присущие ей, либо внешние влияющие факторы, такие как надежность и доступность (включая отказоустойчивость и восстанавливаемость), безопасность (включая обеспечение конфиденциальности и целостность), пригодность для обслуживания, долговечность и техническую поддержку.

Лучше пользоваться определением, которое по смыслу аналогично вышеприведенному, но больше соответствует целям - получению практических навыков создания надежного ПО: «*Надежность ПО* есть вероятность его работы без отказов в течение определенного периода времени, рассчитанная с учетом стоимости для пользователя каждого отказа.»

Принимается во внимание серьезность ошибки, степень ее влияния на работу системы. Следовательно, надежность не является внутренним свойством программы, она во многом связана с тем, как программа

используется. Термин «вероятность» играет в определении значительную роль, указывая на случайный характер проявления ошибки.

Хотя определение надежности ПО сходно с определением надежности аппаратуры, эти понятия должны рассматриваться раздельно. В первую очередь, это определяется различной природой отказов аппаратуры и программ.

У аппаратуры возможны три причины отказа: дефект проектирования; дефект производства; физический износ (старение элементов).

Причина отказа ПО – *исключительно* следствие дефекта (ошибки) проектирования (разработки). *Отказ программного обеспечения* - это проявление ошибки в нем.

Ошибки в ПО не являются его внутренним свойством. Наличие ошибок - функция как самого ПО, так и ожиданий его пользователей. В ПО имеется ошибка, если оно не выполняет того, что пользователю *разумно* от него ожидать: предсказуемого поведения. В данном случае «разумно» означает действие, выполняемое в пределах назначения системы. Можно сказать, что синоним «надежности» - «предсказуемость».

Ошибка ПО проявляется при изменении входных данных. Если подавать на вход ПО одни и те же данные, ошибка не проявится.

На рисунке 24 приведен примерный график зависимости частоты отказов аппаратуры и ПО от времени, иллюстрирующий их разную природу.

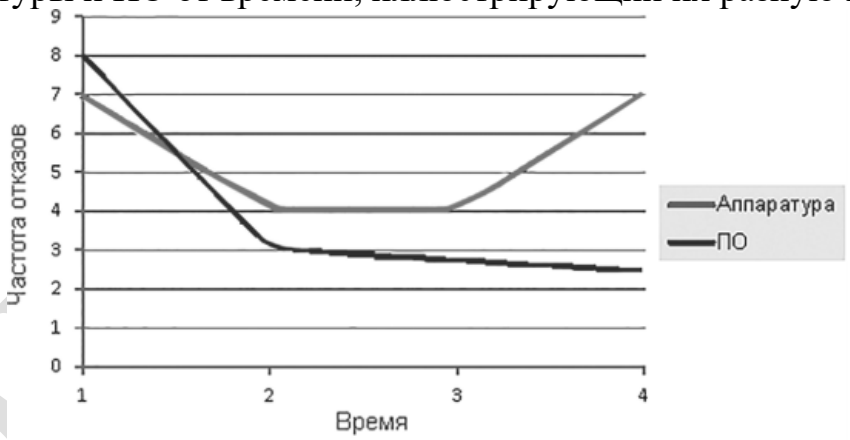


Рисунок 24 – График частоты отказов

График частоты отказов аппаратуры можно условно разделить на три области.

Первая область - область «приработки». В процессе опытной эксплуатации выявляются как дефекты проектирования, так и дефекты производства, включая примененные при изготовлении некачественные элементы.

Вторая область - область стабильной работы. Аппаратура «приработалась», в основном дефекты устранены.

Третья область - область «старения». Частота отказов увеличивается по причине выхода из строя элементов, срок службы которых истек. Традиционно проблема решается путем ремонта и замены отказавших элементов.

Несколько иначе обстоит дело с частотой отказов ПО. Первая область также является областью «приработки». Выявляются и устраняются те

ошибки, которые не были устранены в процессе отладки и испытаний. Далее, если не происходит существенных отклонений от нормальных условий эксплуатации, происходит плавное снижение частоты отказов. Оставшиеся в ПО ошибки не проявляются. Обычная практика - ПО или его версия «выходит в тираж», выводится из эксплуатации с массой не выявленных оставшихся ошибок.

Вывод: невозможно найти все ошибки в достаточно сложном ПО. Но в этом нет ничего страшного. Главное, чтобы они не мешали нормальной работе системы, использующей ПО.

Кривая частоты отказов ПО нарисована из предположений:

- ошибки в ПО исправляются сразу же по мере их обнаружения;
- исправление ошибок не влечет появление новых.

Второе предположение на практике обычно не выполняется. Основное отличие ПО от аппаратуры график отражает: программы не стареют и не изнашиваются. Только ошибки являются причиной его отказов.

2 Загрузка и установка программного обеспечения

2.1 Понятие совместимости программного обеспечения. Аппаратная и программная совместимость. Совместимость драйверов

Совместимость (compatibility) — способность аппаратных или программных средств работать с компьютерной системой. Аппаратная (техническая) совместимость (hardware (equipment) compatibility) — способность одного компьютера работать с узлами или устройствами, входящими в состав другого компьютера. Составной частью аппаратной совместимости является электромагнитная совместимость (ЭМС) (ElectroMagnetic Compatibility, EMC) — способность работающих (в том числе, автономно друг от друга) технических средств не создавать взаимных электромагнитных помех, а также функционировать при наличии внешних электромагнитных полей. Также ЭМС называют ограничение собственного электромагнитного излучения устройств до уровня, не влияющего на работу других устройств.

Совместимость (compatibility) — способность аппаратных или программных средств работать с компьютерной системой. Аппаратная (техническая) совместимость (hardware (equipment) compatibility) — способность одного компьютера работать с узлами или устройствами, входящими в состав другого компьютера. Составной частью аппаратной совместимости является электромагнитная совместимость (ЭМС) (ElectroMagnetic Compatibility, EMC) — способность работающих (в том числе, автономно друг от друга) технических средств не создавать взаимных электромагнитных помех, а также функционировать при наличии внешних электромагнитных полей. Также ЭМС называют ограничение собственного электромагнитного излучения устройств до уровня, не влияющего на работу других устройств. *Совместимость* (compatibility) — способность аппаратных или программных средств работать с компьютерной системой. Аппаратная (техническая) совместимость (hardware (equipment) compatibility) — способность одного компьютера работать с узлами или устройствами, входящими в состав другого компьютера. Составной частью аппаратной совместимости является электромагнитная совместимость (ЭМС) (ElectroMagnetic Compatibility, EMC) — способность работающих (в том числе, автономно друг от друга) технических средств не создавать взаимных электромагнитных помех, а также функционировать при наличии внешних электромагнитных полей. Также ЭМС называют ограничение собственного электромагнитного излучения устройств до уровня, не влияющего на работу других устройств.

Информационная совместимость (data compatibility) — способность двух или более компьютеров или систем адекватно воспринимать одинаково представленные данные. Частью информационной совместимости, а также средством ее обеспечения является совместимость форматов представления данных.

Программная совместимость (software compatibility) — возможность выполнения одних и тех же программ на разных компьютерах с получением одинаковых результатов (не путать с совместимостью программ).

Совместимость программ (program compatibility) — пригодность программ к взаимодействию друг с другом и, в частности, к объединению в программные комплексы для решения более сложных задач, например, в автоматизированных системах.

Полная совместимость (fully compatibility) — аппаратная, программная и информационная совместимость двух или более компьютеров без каких-либо ограничений для их пользователей.

Совместимость компьютеров определяют, как правило, по отношению к компьютерам IBM AT. Как уже было сказано, первые ПК класса IBM AT создавались из уже широко представленных на компьютерном рынке элементов. Любой инженер, имеющий представление о структуре вычислительной системы, может без труда из таких "покупных" деталей собрать свой собственный ПК, подобный IBM AT. Единственным запатентованным фирмой IBM компонентом является набор из двух микросхем, названный BIOS (базовая система ввода- вывода). Именно записанный в этих схемах код лежит в основе совместимости.

В настоящее время программная совместимость уже не является камнем преткновения. Наборы BIOS даже самых малоизвестных фирм обеспечивают совместимость с компьютерами фирмы IBM. Разработчики программного обеспечения, стремясь расширить использование своей продукции, обязательно должны учитывать программную совместимость с компьютерами фирмы IBM.

Аппаратная совместимость. Этот термин относится к системам, допускающим сопряжение, или устройствам с взаимозаменяемыми конструктивными узлами. Употребляют его и в случае, когда речь идет о модулях расширения. Некоторые из них, такие как платы памяти или видеоадаптер, не могут работать с набором BIOS некоторых ПК. В этом случае либо применяют другой набор BIOS, либо заменяют плату расширения.

Однако современные производители аппаратуры, стараясь заполучить как можно больше покупателей, стремятся во что бы то ни стало обеспечить аппаратную совместимость своей продукции с выпускаемыми изделиями.

Совместимость внутри одного пакета программ. Иногда случается так, что программы одного и того же программного обеспечения не могут вместе "сосуществовать" независимо от типа компьютера. Наиболее часто такие "конфликты" возникают между резидентными программами. Такая несовместимость не оказывает влияния на работу самого ПК, нарушается лишь выполнение программ.

Совместимость плат расширения. Иногда несовместимыми могут оказаться платы расширения. При этом не нарушается работа всего компьютера. Такая несовместимость связана лишь с невозможностью одновременной работы двух или более плат. В этом случае необходимо заменить одну из них.

Скоростная совместимость. Выполнение некоторых программ возможно лишь при определенной скорости. В случае запуска таких программ на более быстром компьютере их выполнение нарушается. Исправить подобное положение можно либо переключением на более низкую скорость, либо использованием сервисных программ, вызывающих снижение скорости работы.

Следует заметить, что два компьютера, IBM AT и IBM XT286, имеют наиболее высокую аппаратную и программную совместимости. Поэтому именно их приняли за стандарт, относительно которого определяется совместимость. Так, компьютеры типа IBM PS/2 Model 50 и 60, имея программное обеспечение, совместимое с IBM AT, физически с ними несовместимы. Это обусловлено принципиально новой аппаратной частью систем IBM PS/2. Их совместимость с ПК семейства AT возрастает по мере разработки новых стандартов на аппаратуру PS/2 и создания большего числа модулей расширения для них.

Совместимость аналогов и так называемых совместимых компьютеров с "фирменными" ПК постоянно повышается. В отличие от аналогов IBM-совместимые не являются полностью взаимозаменяемыми с самими компьютерами IBM AT. Отсюда и низкая оценка их аппаратной совместимости.

Несовместимость того или иного рода возможна в любом компьютере. Причиной ее возникновения, как было сказано, является отклонение от стандартного набора BIOS. Поэтому чем ближе набор BIOS компьютера к оригиналу AT, тем выше совместимость.

Программная совместимость ПК обеспечивается в первую очередь применением в них одной и той же операционной системы или однотипных операционных систем.

Стремление добиться совместимости компьютеров, выпускаемых разными фирмами в разных странах, привело к появлению в мире особого класса ПК, так называемых «IBM-совместимых».

В этих компьютерах используются микропроцессоры и операционные системы, являющиеся аналогами микропроцессоров и операционных систем, используемых в моделях ПК фирмы IBM.

В составе ЦВМ (цифровая вычислительная машина) в соответствии с определением вычислительной машины выделяют ряд устройств.

Устройство — часть машины, имеющая определенное функциональное назначение.

В соответствии с принципами построения и действия ЦВМ в состав любой ЦВМ входят:

– арифметико-логическое устройство (АЛУ) — функциональная часть ЦВМ, предназначенная для выполнения операций преобразования (обработки) величин: арифметических, логических (поразрядных), сдвига;

– устройство может включать в себя один универсальный операционный блок, настраиваемый на выполнение требуемой операции, или несколько операционных блоков, предназначенных для выполнения разнотипных операций;

– запоминающее устройство (ЗУ) — функциональная часть ЦВМ, предназначенная для записи, хранения и выдачи информации, представленной цифровыми кодами.

Довольно часто такое отдельное устройство называют памятью, т.е. слова «запоминающее устройство» и «память» — синонимы.

Совместимость драйверов

Драйверная концепция – неотъемлемая часть современных операционных систем. Эта концепция – основа взаимодействия системы (пользователя) с какими бы то ни было устройствами (системными/периферийными, реальными/виртуальными и т.д.). Даже системные программисты далеко не всегда имеют представление об этой концепции, о принципах ее работы, о программировании с использованием этой концепции. А ведь системное программирование – ключ к пониманию основ IT.

Написание драйверов – достаточно сложная, но, тем не менее, очень интересная и актуальная отрасль программирования. Знание особенностей технологии написания драйверов открывает огромное количество возможностей – написание драйверов для устройств, уже не поддерживаемых производителем, для устройств, драйвера к которым еще не написаны, исправление ошибок в драйверах, написание драйверов к различным промышленным устройствам и т.д.

У каждой операционной системы есть свои особенности, отсюда вытекает своя специфика написания драйверов под них. То же самое можно сказать и о разных типах оборудования.

Разработкой драйверов обычно занимаются профессионалы, каждая крупная компания, выпускающая технику имеет целый штат сотрудников, занимающихся разработкой драйверов. Прежде всего, разработчик драйвера должен владеть программированием на языке C (без расширений C++), поскольку описание синтаксиса и применения конструкций этого языка не рассматриваются в данной книге вовсе. Во-вторых, разработчик драйверов, пусть начинающий, должен иметь твердо сформировавшееся представление о программировании в многозадачной среде при интенсивном использовании многопоточности.

Драйвер – это часть кода операционной системы (небольшая программка не для пользователя, а для ОС), отвечающая за взаимодействие с аппаратурой. Под словом «аппаратура» можно подразумевать как реальные физические устройства, так и виртуальные и логические. Например, есть много разных, принтеров разных производителей. Чтобы на них печатать, нужно знать, какие команды этому принтеру нужно передавать. У разных производителей такие команды могут сильно отличаться друг от друга. Нужно как-то научить ОС работать со всеми типами принтеров – в этом помогают драйверы.

С момента своего появления и до сегодняшнего дня драйвер непрерывно эволюционировал, и процесс этот продолжается до сих пор. Один из моментов эволюции драйвера – это эволюция концепции драйвера, как легко заменяемой части операционной системы. Как отдельный и

довольно независимый модуль драйвер сформировался не сразу, да и сейчас многие драйверы практически неотделимы от операционной системы. Во многих случаях это приводит к необходимости переустановки системы (ОС Windows) или переборки ее ядра (в UNIX-системах).

Список основных общих концепций драйверов в Windows и UNIX-системах выглядит так:

1. Способ работы с драйверами как файлами. Т.е. функции, используемые при взаимодействии с файлами, практически идентичны таковым при взаимодействии с драйверами (лексически): open, close, read и т.д.
2. Драйвер, как легко заменяемая часть ОС
3. Существование режима ядра

Классификация типов драйверов для ОС Windows:

1. Драйверы пользовательского режима (User-Mode Drivers):

- Драйверы виртуальных устройств (Virtual Device Drivers, VDD) – используются для поддержки программ MS-DOS;
- Драйверы принтеров (Printer Drivers);

2. Драйверы режима ядра (Kernel-Mode Drivers):

- Драйверы файловой системы (File System Drivers) – осуществляют ввод/вывод на локальные и сетевые диски
- Унаследованные драйверы (Legacy Drivers) – написаны для предыдущих версий Windows
- Драйверы видеоадаптеров (Video Drivers) – реализуют графические операции
- Драйверы потоков устройств (Streaming Drivers) – осуществляют ввод/вывод потоков видео и звука
- WDM-драйверы (Windows Driver Model) – поддерживают технологию Plug and Play и управления электропитанием.

Драйверы бывают одно- и многоуровневыми. Если драйвер является многоуровневым, то обработка запросов ввода/вывода распределяется между несколькими драйверами, каждый из которых выполняет свою часть работы. Между этими драйверами можно поставить любое количество фильтр-драйверов (filter-drivers). При обработке запроса данные идут от вышестоящих драйверов к нижестоящим, а при возврате – наоборот. Одноуровневый драйвер (monolithic) драйвер является противоположностью многоуровневому.

Для технологии Plug and Play существуют три уровня-типа драйверов:

- шинные драйверы
- фильтр-драйверы
- функциональные драйверы

Системное ПО, поддерживающее Plug and Play предоставляет следующие возможности:

- автоматическое распознавание подключенных к системе устройств
- распределение и перераспределение ресурсов (таких, как порты ввода/вывода и участки памяти) между запросившими их устройствами
- загрузка необходимых драйверов

- предоставление драйверам необходимого интерфейса для взаимодействия с технологией Plug and Play

- реализация механизма, позволяющего драйверам и приложениям получать информацию касаемо изменений в наборе устройств, подключенных к системе устройств, и совершить необходимые действия

- предоставление драйверам необходимого интерфейса для взаимодействия с технологией Plug and Play

- реализация механизма, позволяющего драйверам и приложениям получать информацию касаемо изменений в наборе устройств, подключенных к системе устройств, и совершать необходимые действия.

Инструментарий:

Существует много утилит, которые могут понадобиться при разработке драйверов. Основным средством разработки является Microsoft Windows DDK, Device Driver Kit, — пакет разработки драйверов, включающий компилятор, редактор связей (линкер), заголовочные файлы, библиотеки, большой набор примеров (часть из которых является драйверами, реально работающими в операционной системе) и, разумеется, документацию. В состав пакета входит также отладчик WinDbg, позволяющий проводить интерактивную отладку драйвера на двухкомпьютерной конфигурации и при наличии файлов отладочных идентификаторов операционной системы WinDbg кроме того, позволяет просматривать файлы дампа (образа) памяти, полученного при фатальных сбоях операционной системы (так называемый crash dump file).

Языком программирования, который используется в DDK является язык C, допускающий вставки на языке ассемблер, который в былые времена был основным и единственным языком программирования драйверов.

В бесплатно распространяемом пакете DDK всегда отсутствовала интегрированная среда разработки. Поэтому есть необходимость использовать Visual Studio в качестве средства редактирования исходного кода. При должной настройке этой среды процесс выявлений синтаксических ошибок существенно облегчается — неотъемлемое преимущество интегрированных сред программирования. Компилятор и редактор связей Visual Studio C++ создают нормальный бинарный код, вполне работоспособный при указании соответствующих опций (настроек) компиляции, однако эталоном следует считать бинарный код, получающийся при компиляции кода драйвера с использованием утилиты Build из состава пакета DDK. Разумеется, встроенный интерактивный отладчик Visual Studio и прилагаемая документация становятся для разработки драйвера совершенно бесполезными, поскольку не предназначены для работы с программным обеспечением для режима ядра.

Есть пакеты разработки драйверов и от других фирм: WinDriver или NuMega Driver Studio, но у них есть отличия базиса функций Microsoft (порой довольно большие) и масса других мелких неудобств.

Так же необходимо использовать некоторые сторонние утилиты, такие как: редактор реестра, мониторы обращений к реестру и файлам, средство

просмотра каталогов имен объектов, программы просмотра, сохранения и т.д. отладочных сообщений.

2.2 Причины возникновения проблем совместимости. Методы выявления проблем совместимости ПО

Программная неисправность компьютера встречается гораздо чаще, чем аппаратная. Причины подобного рода ошибок:

Несовершенство ПО (программного обеспечения). Обычный разработчик создает ПО работающее с определенными ресурсами. Профессиональный программист создает ПО, адаптируя его для работы в разных условиях.

Программа распоряжается системными библиотеками ОС. При обновлении т.е. улучшении версии системных библиотек, программа или несколько программ могут отказаться работать.

Несовершенство ОС (операционных систем). ОС не могут поддерживать работу всего существующего ПО. И разработчики пишут ПО под конкретную ОС. Т.е. пользователь работает либо с сертифицированными программами, либо мириться с ошибками, возникающими из-за проблем несовместимости ПО.

Отсутствие нужных ресурсов. Часто небольшая программа требует для нормальной работы ресурсов больше, чем предоставляет ОС. Для обеспечения работы программы ОС увеличивает файл подкачки (файл на жестком диске, используемый для организации виртуальной памяти; виртуальная память – расширенное адресное пространство задачи, полученное за счет использования части внешней памяти; компьютерная проблема – нехватка оперативной памяти; в оперативной памяти всегда находиться часть виртуального пространства, остальная его часть располагается на дисковой памяти; при нехватки оперативной памяти, неиспользуемое приложение или его часть выгружается из ОП на диск и загружается необходимое активное приложение), отбирая ОП у других программ.

Это приводит к снижению производительности ПК (персональный компьютер) и ошибкам.

Проблему можно решить простым увеличением объема оперативной памяти или с помощью специальных утилит.

Ошибки в реестре. Реестр – мозг ОС. Ошибки в нем сказываются на всех процессах, происходящих в ПК. Причиной сбоя в реестре непрофессионально написанные программы, прописывающие свои файлы и ссылки в самых разных местах.

Не стоит забывать и о вирусах и вредоносном ПО, которое портит файлы, ссылки на них.

Для «лечения» реестра предназначены специальные утилиты, умеющие анализировать записи и удалять ошибочные и неиспользуемые данные, не забывайте об элементарном сохранении рабочей версии файлов реестра.

Конфликты между устройствами. Аппаратное обеспечение не бывает полностью совместимо и может отказаться работать. Устранить проблему неисправности звуковой карты или привода компакт-диска с помощью диспетчера устройств. Либо проанализировать инструкцию к устройству,

возможно, вы найдете информацию о проблемах несовместимости. Изучите всю прилагаемую документацию. Совместимость восстанавливается если:

- Переустановить плату расширения в другой слот;
- Заменить устройство более новой моделью.

Вирусы, троянские кони и «черви». Все вирусы (компьютерный вирус – это небольшая программа, способная создавать свои копии и внедрять их в различные программные объекты компьютера без ведома пользователя; при этом копии сохраняют способность дальнейшего размножения; если вирус «внедрился» в файл, то такой файл считается «зараженным»; вирусы могут заражать область системного загрузчика, исполняемые файлы, файлы драйверов, документы Word, Excel и т.д.) могут попасть на ПК через интернет, зараженный флэш - носитель. Защитить ПК поможет хорошая антивирусная программа и настроенный брандмауэр. Либо при невозможности устранить заражение ПК стоит переустановить ОС. Программные вирусы классифицируются по среде обитания: сетевые, файловые, загрузочные, файлово - загрузочные.

Проверка ПК на вирусы: подавляющее большинство зараженных файлов можно «вылечить». Существует множество программ (антивирусов) для обнаружения вирусов и лечение файлов. Программа Doctor Web содержит алгоритм, позволяющий обнаружить и неизвестные вирусы. Программа предназначена для работы в среде Windows и имеет удобный пользовательский интерфейс. Задание параметров тестирования осуществляется с помощью системного меню. При запуске программа проверяет оперативную память на наличие вирусов и обезвреживает их. В программе можно получить следующие сведения: вывод отчета об обнаруженных зараженных файлах; лечение зараженных файлов; удаление найденных зараженных файлов; переименование зараженных файлов; перемещение зараженных файлов. Для того чтобы потери от вируса были минимальными, рекомендуется создавать незараженные копии файлов используемой информации на CD-дисках или других запоминающих устройствах. При переносе файлов с компьютера на компьютер следует обязательно проверять на наличие вирусов и лечить зараженные файлы. Периодически обновляйте версии антивирусной программы, т.к. постоянно создаются новые компьютерные вирусы.

Ограничения ОС. ОС обеспечивает свою безопасность посредством ограничений:

- На установку новых программ;
- На удаление файлов;
- На просмотр веб - ресурсов;
- И т.д.

Решение проблемы, найти программу не вызывающую у ОС такие типы ограничений, либо обновить ОС.

Использование устаревшего оборудования. После обновления ОС ваше оборудование может выдавать ошибки. Выход – обновить драйвера устройств или заменить оборудование современными моделями.

Неверные настройки ОС

Распространенные неполадки:

- неверное разрешение или мерцание экрана;
- исчезновение языков;
- отсутствие или искажение звука;
- низкая скорость модема;
- проблемы с локальной сетью
- и т.п.

ОС будет работать так, как вы ее настроили!!!

Настройка системы через панель управления, менеджеры устройств.

Решение проблемы: снять неверные настройки видеодисплея, можно загрузив ПК в *безопасном режиме* (или «чистая загрузка») и исправив настройки.

Сразу после идентификации жестких дисков, нажмите клавишу F8 или CTRL или другая клавиша. Результатом станет меню:

1. Normal – обычная загрузка Windows
2. Logger (\bootlog.txt) – загрузка с протоколированием в файле
3. Safe mode – режим защиты от сбоев
4. Step-by-step confirmation – пошаговая загрузка
5. Command prompt only – режим командной строки
6. Safe mode command prompt only – защищенный режим командной строки

В режиме защиты от сбоев произойдет загрузка с использованием минимального комплекта драйверов устройств. После загрузки можно изменить настройки, которые вызвали сбой. После завершите работу обычным образом и перезагрузитесь.

Определение того, когда следует использовать исправление совместимости

Решение о том, как использовать исправления совместимости для устранения проблем с совместимостью, может потребовать больше, чем просто технические проблемы. Следующие сценарии отражают другие распространенные причины использования исправления совместимости.

Сценарий 1. В приложении, которое больше не поддерживается поставщиком, существует ошибка совместимости.

Как и во многих компаниях, вы можете запускать приложения, для которых поставщик завершил поддержку. В этом случае поставщик не может внести в него исправления, и вы не можете получить доступ к исходному коду для самостоятельного изменения проблемы. Однако возможно, что использование исправления совместимости может решить проблему совместимости.

Сценарий 2. В приложении, созданном внутренним приложением, существует ошибка совместимости.

Для устранения проблемы предпочтительнее исправить код приложения, но это не всегда возможно. Возможно, ваша внутренняя группа не может решить все проблемы до развертывания новой операционной системы. Вместо этого они могут использовать исправление совместимости везде, где это возможно. Затем они смогут исправить этот код только для

проблем, которые невозможно устранить таким образом. С помощью этого метода ваша команда может изменить приложение так, как можно допустить время, не откладывая развертывание новой операционной системы в вашу среду.

Сценарий 3. В приложении, для которого должна быть выпущена совместимая версия в ближайшем будущем, или в приложении, которое не является критическим для организации независимо от ее версии, существует ошибка совместимости.

В ситуации, когда приложение не имеет значения для вашей организации, или для более поздней версии, которая вскоре будет выпущена, вы можете использовать исправление совместимости в качестве временного решения. Это означает, что вы можете продолжать использовать приложение, не откладывая развертывание новой операционной системы, предоставляя возможность обновлять конфигурацию, как только новая версия будет выпущена.

2.3 Выполнение чистой загрузки. Выявление причин возникновения проблем совместимости ПО. Выбор методов выявления совместимости

Чистая загрузка в системе Windows 7, 8.1 и 10

Если возникли проблемы с компьютером или программным обеспечением, то в поисках решения в сети Интернет можете наткнуться на чьи-то советы, в которых автор рекомендует выполнить так называемую чистую загрузку. Очень часто многие рекомендуют ее на официальных игровых форумах. Если игра или программа медленно, или нестабильно работает либо самостоятельно закрывается, то часто пользователи получают совет выполнить чистую загрузку.

Чистая загрузка – это специальный метод запуска системы таким образом, чтобы во время запуска не включались дополнительные службы и сторонние программы (не относящиеся к Microsoft). В общем, она представляет собой чистый запуск Windows без лишних дополнений и сторонних программ. Но это не безопасный режим – в этом случае система загружается в обычном режиме, но без приложений и служб сторонних производителей.

При запуске Windows таким способом:

- не загружаются ненужные службы;
- активируются только компоненты Microsoft;
- игнорируются программы из автозагрузки.

Очень часто проблемы с производительностью в играх или со стабильностью работы программ возникают из-за конфликтов между программным обеспечением. Одна небольшая программа, например, которая добавляет новые функции для клавиатуры или мыши в состоянии негативно повлиять на какую-то игру, из-за чего та начнет самостоятельно закрываться, зависать или вообще перестает запускаться. Одна служба, добавленная

внешним программным обеспечением, может вызвать проблемы, препятствующие запуску некоторых приложений.

Но пока не убедитесь однозначно, что проблемы вызывает стороннее ПО, в том числе те, которые чрезмерно загружают оперативную память в компьютере, то не будете точно знать, где искать решение и как устранить ошибку. Чистая загрузка в состоянии подтвердить (или опровергнуть) то, что источником проблемы является стороннее программное обеспечение, установленное в системе.

В общем, благодаря загрузке таким способом можем убедиться, что проблемы с компьютером, играми и приложениями вызывают другие программы, установленные в системе. Если после проблемы с компьютером исчезают, это значит, что какие-то программы, не являющиеся по умолчанию компонентами Windows, вызывают конфликты. В этом случае уже известно, где искать проблемы и можно начинать по одному отключать или удалять их с компьютера, пока не найдете ту, которая вызывает ошибки.

Для непосредственной чистой загрузки в Windows необходимо:

– нажать сочетание клавиш Windows + R для вызова окна «Выполнить». Затем ввести команду msconfig и подтвердить ее запуск;

– отобразится окно конфигурации системы. Перейти в закладку «Службы». Здесь находится список всех системных служб, которые запускаются вместе с операционной системой;

– в нижней части отметить поле «Не отображать службы Microsoft». Благодаря этому в списке останутся только те, которые были добавлены сторонним программным обеспечением во время установки на компьютере (Когда отображаются только службы, не являющиеся частью Windows, кликнуть на кнопку «Отключить все»).

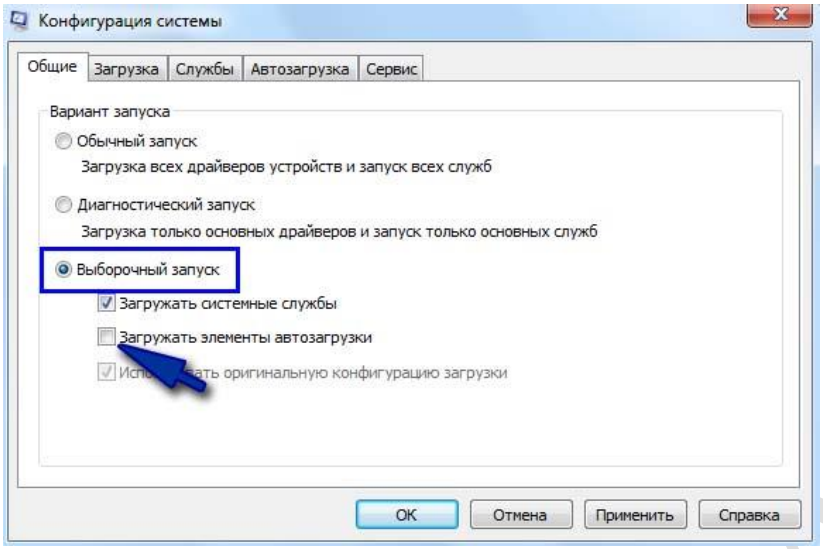
– затем перейти на вкладку «Автозагрузка» и нажать на кнопку «Открыть Диспетчер задач». Здесь в списке отображаются все программы, которые запускаются вместе с системой. Нажать на каждую из них правой кнопкой мыши и выбрать «Отключить». Для чистой загрузки нужно их все отключить.

– нажать кнопку ОК в Диспетчере задач, чтобы подтвердить настройки. Откроется окно Настройки системы. Здесь также нажать «Применить» и «ОК», чтобы закрыть окно, и сохранить изменения. Появится запрос, применить изменения при перезапуске – выбрать вариант сохранить изменения и перезапустить.

Теперь компьютер после перезагрузки запустится в режиме чистой загрузки, то есть без сторонних компонентов.

Чистая загрузка Windows 7 настраивается подобным образом. Здесь также нажать сочетание клавиш Windows + R, чтобы вызвать окно «Выполнить» и наберите команду: msconfig

Появится окно Конфигурации системы. На вкладке «Общие» отметить пункт «Выборочный запуск». Прямо под ним убрать отметку с пункта «Загружать элементы автозагрузки». Таким образом, за один раз отключен автозапуск сторонних программ.



Затем открыть закладку Службы. Здесь отметить пункт «Не отображать службы Майкрософта», а после нажать на кнопку «Отключить все». Таким образом, будут отключены те, которые не являются неотъемлемой частью Windows и могут вызывать ошибки.

При закрытии настроек появится сообщение – подтвердите сохранение изменений и перезапустите компьютер. Система перезагрузится в режиме чистой загрузки.

Вывод***:*** Если удалось исправить проблему, то вывод один: какая-то программа вызывает ошибки. Теперь можно по очереди включать системные службы в окне msconfig и программы в автозагрузке, и после каждого включения перезагружайте компьютер. Когда найдутся несовместимое ПО, то самый простой способ – удалить его или обновить до последней версии.

Использование средства анализатора доступа учетных записей

С помощью средства Standard User Analyzer (SUA) можно тестировать приложения и отслеживать вызовы API для выявления проблем совместимости, связанных с функцией контроля учетных записей.

Мастер SUA также устраняет проблемы, связанные с UAC. В отличие от средства SUA, мастер SUA поможет Вам пошагово выполнить процесс, не прибегая к детальному анализу средства SUA. Дополнительные сведения о мастере SUA можно найти в разделе Использование мастера SUA.

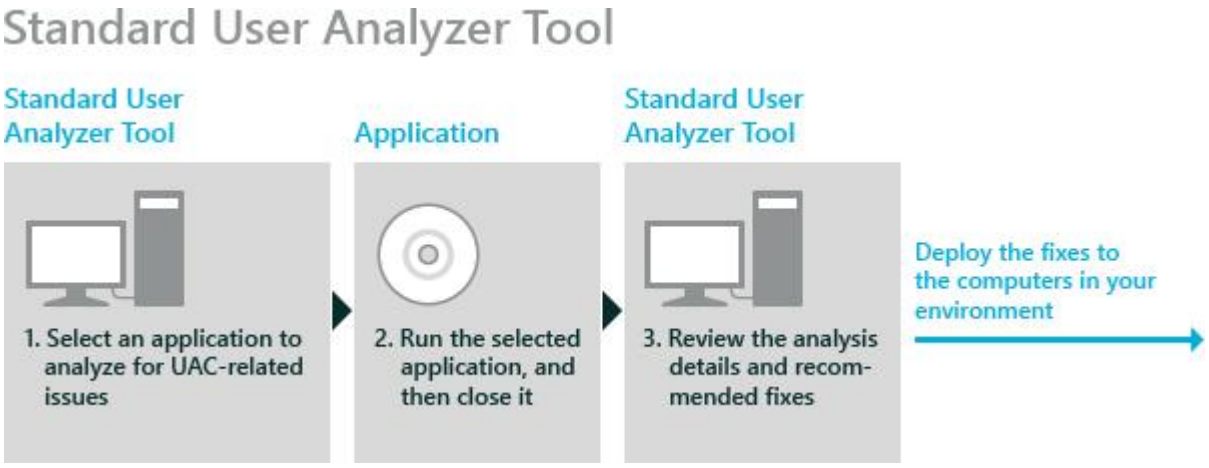
В средстве SUA вы можете включать и отключать виртуализацию. Если вы отключаете виртуализацию, тестируемое приложение может работать в более похожем образом в более ранних версиях Windows®.

В инструменте "SUA" можно выбрать Запуск приложения с *правами администратора* или *обычного пользователя*. В зависимости от сделанного выбора вы можете найти различные типы проблем, связанных с UAC.

Тестирование приложения с помощью средства SUA

Прежде чем можно будет использовать средство SUA, необходимо установить средство проверки приложений. Кроме того, необходимо установить Microsoft® .NET Framework 3,5 или более поздней версии.

В следующей блок-схеме показан процесс использования средства SUA.



Сбор проблем, связанных с UAC, с помощью средства "SUA"

1. Закройте все открытые экземпляры мастера служебных программ или SUA на компьютере.

Если на компьютере уже есть экземпляр SUA, средство SUA откроется в режиме просмотра журнала, а не в обычном режиме. В режиме просмотра журнала нельзя запускать приложения, что предотвращает сбор проблем UAC.

2. Запустите анализатор Standard User Analyzer.

3. В поле *Target приложение* найти исполняемый файл приложения, которое вы хотите проанализировать, и дважды щелкн его, чтобы выделить.

4. Снять флажок *повышать уровень* и нажать кнопку *запустить*.

Если появится диалоговое окно *разрешение отвергнуто*, нажать кнопку *ОК*. Приложение будет запущено, несмотря на предупреждение.

5. Проучить аспекты приложения, для которого требуется собрать сведения о проблемах UAC.

6. Выйти из приложения.

7. Ознакомиться со сведениями на разных вкладках в средстве "SUA". Сведения о каждой вкладке можно найти в разделе вкладки интерфейса средства SUA.

Чтобы просмотреть и применить рекомендуемые способы устранения проблем

1. В средстве "SUA" в меню "*снижение рисков*" выбрать команду "*применить меры по снижению риска*".

2. Ознакомиться с рекомендациями по устранению проблем с совместимостью.

3. Нажать кнопку *Применить*.

Средство SUA создает пользовательскую базу данных исправления совместимости и автоматически применяет ее к локальному компьютеру, чтобы можно было протестировать эти исправления, чтобы убедиться, что они работают.

2.4 Проблемы перехода на новые версии программ. Мастер совместимости программ. Инструментарий учета аппаратных компонентов

Основные этапы перехода на последнюю версию ПО:

1. *Апгрейд (Upgrade).* На данном этапе происходит обмен одного из имеющихся у пользователя продукта на новый. Это предложение действует только для обладателей лицензионной системы. «Старая» программа остается у клиента и ею можно пользоваться во время «переходного» периода.

2. *Установка*. Предусматривает проведение самих работ, результат – рабочая новая версия ПО, которую можно сразу же использовать.

3. *Перенос данных*. Необходимый этап в рамках, которого информация со старой версии переносится на новую.

4. *Освоение*.

Научиться работать с новой системой можно:

– самостоятельно, для этого выпускается специализированная литература;

– в специально созданных учебных центрах;

– индивидуально, с квалифицированным специалистом.

Несмотря на все преимущества новой версии, многие фирмы продолжают работать на базе старой.

Вопрос перехода с предыдущего ПО или ОС на более позднюю версию становится все более актуальным для компаний, до сих пор использующих это ПО в своей работе.

НАПРИМЕР : Срок действия поддержки Windows XP истек 14 апреля 2014 года, после чего ИТ-отделам нужно было оперативно переходить на Windows 7 или Windows 8.1, чтобы избежать проблем с обеспечением безопасности и несоответствия требованиям.

Стандартная процедура модернизации может занять не один месяц, а этого времени у организаций просто нет. Не менее важно, что ИТ-отделы не могут допустить расходования ресурсов на проекты ПО и приостановления ключевых процессов для поддержки мобильности предприятия.

Следовательно, необходимо решение, которое позволит осуществить миграцию со старого ПО, а также сделать шаг вперед и обеспечить условия для продуктивной работы современного штата сотрудников в любое время, в любом месте и с любого устройства.

Для перехода к более поздней версии ПО и трансформации предприятия ИТ-отделам и организации необходимо провести:

– сокращение риска благодаря устранению опасных пробелов в системе безопасности и несоответствий, возникающих в результате продолжительного использования старого ПО;

– увеличение продуктивности ИТ-отделов благодаря использованию виртуальных десктопов для перехода на последнее ПО;

– экономия за счет доставки виртуальных приложений и десктопов на существующее аппаратное обеспечение;

– модернизация среды и предоставление сотрудникам возможности достигать большей эффективности при работе в любое время, в любом месте и на любом устройстве.

Безопасность

Окончание поддержки Windows также означает прекращение выпуска обновлений безопасности. *ПК, которые на момент окончания поддержки продолжат использовать эту ОС, окажутся гораздо более уязвимыми к угрозам, вирусам, шпионскому ПО и другим вредоносным действиям.*

Всего одна атака может подвергнуть опасному риску интеллектуальную собственность организации, данные клиентов, важную персональную информацию, повседневную деятельность и на долгое время привести к возбуждению исков и ущербу для репутации компании.

Риск является существенным уже сейчас: по данным Microsoft, например, вероятность заражения Windows XP вредоносными программами была в 21 раз выше вероятности заражения Windows 8, и эта цифра резко увеличится после того, как обнаруженные уязвимости в системе безопасности перестанут защищаться дополнениями или устраняться. В связи с повторным использованием кода в Windows XP и Windows 7/8.1, дополнения к последним версиям Windows могут привести к выявлению в Windows XP пробелов в системе безопасности, что позволит хакерам манипулировать компаниями, по-прежнему использующими эту платформу.

Соответствие стандартам

Как внутренние, так и внешние аудиторские проверки будут требовать от предприятий использования версий ПО с действующей поддержкой и со всеми установленными обновлениями; разумеется, это базовое требование, которое должна быть в состоянии выполнить любая организация.

Несоблюдение этих стандартов может легко привести к ущербу для бизнеса, повышенным ставкам за страхование предпринимательской деятельности и даже пробелам в системе страхового обеспечения бизнеса.

Последствия несогласованности могут оказаться особенно серьезными в отраслях, подвергаемых специальному надзору, таких как здравоохранение и финансовые службы, что приведет к утрате разрешений контролирующих органов.

Организации со сторонними поставщиками услуг и выполняемыми сторонними организациями контрактами также могут столкнуться с проблемой отказа этих поставщиков от соглашений в отношении поддержки десктопов для ПО, что приведет к еще большим трудностям.

Увеличивающийся уровень сложности:

– Окончание поддержки выявит массу вытекающих одна из другой проблем, не ограничивающихся вопросами безопасности и соответствия нормативным требованиям.

– Многочисленные зависимости от платформ и технологии других компаний, обычно используемые в среде предприятия, потенциально

усложняют предстоящий переход к более поздним версиям ПО, а уровень сложности будет возрастать по мере переноса этих элементов инфраструктуры в среды, несовместимые с другим ПО.

– Необходимость разрешения этих проблем, а также проблем, связанных с вопросами *безопасности и совместимости*, будет делать работу ИТ-отделов все более и более сложной и времязатратной и отнимать как кадровые, так и финансовые ресурсы, которые можно было бы направить на более стратегически важные задачи.

Деятельность ИТ-отдела может быть парализована как раз в моменты возникновения новых, наиболее приоритетных задач, среди которых обеспечение мобильности организации и ориентации на потребителя.

Переход к более поздней версии ПО является вопросом как осуществить его наилучшим образом. Для большинства организаций модернизация ПО является крупнейшим за многие годы мероприятием, касающимся КС.

Проект может потребовать большого бюджета, ресурсов и времени и привести к возникновению серьезных проблем как для ИТ-отделов, так и для пользователей.

Исключить проблемы, связанные с совместимостью приложений. Многие организации используют важнейшие для них приложения, которые несовместимы с версиями ПО более новыми, что создает серьезные препятствия для модернизации ОС. Более того, вы можете не знать, сколько таких приложений может быть или к какому типу они относятся.

Нужно определять место каждого приложения с точки зрения совместимости, прежде чем сделать шаг вперед. Пользователи смогут без проблем получать доступ как к устаревшим приложениям, так и к обновленным приложениям в течение всего процесса.

Обеспечить защиту пользовательских данных в кратчайшие сроки. Пока данные остаются на конечных устройствах на базе старой ОС, организации ежедневно подвергаются риску нарушения безопасности, воздействия вредоносных программ, несоответствия нормативным требованиям и наложения штрафов контролирующими органами. Даже модернизированный ПК или мобильное устройство подвержены риску утери, кражи или перехвата интеллектуальной собственности, данных клиента или важной личной информации. ОС без действующей поддержки увеличивает такую уязвимость в несколько раз.

Доставлять приложения на существующее аппаратное обеспечение и сокращать расходы**.** Как и в случае с любой другой модернизированной ОС, для эффективного функционирования нового ПО может потребоваться и модернизация конечных устройств пользователей. Это предполагает дополнительные капитальные и операционные расходы и создает новые трудности, касающиеся логистики. *Чтобы не поддерживать практически бесконечное разнообразие пользовательских конфигураций (каждая из которых имеет свой собственный набор потенциальных конфликтов и*

сбоев), необходимо использовать единый стандартный образ для поддержки сотрудников вне зависимости от используемого ими устройства.

От миграции к трансформации**.** После решения вопроса перехода к более поздней версии ПО, важно обратить внимание на главные стратегические инициативы, касающиеся мобильности предприятия, и добиться прогресса в этом направлении. *Это позволит решить задачи, связанные с переходом к более поздней версии ПО, и трансформировать вашу среду в соответствии с актуальными потребностями в рамках одного и того же решения.*

Предоставить современному штату сотрудников возможность эффективно работать в любое время и в любом месте**.** Сегодняшним пользователям необходима возможность эффективно работать в любом нужном им месте и в любое время. Организациям необходим действенный способ, позволяющий поддерживать гибкий формат работы и другие программы обеспечения мобильности, а также обслуживать удаленных работников и сотрудников филиалов.

Предоставьте сотрудникам возможность использовать любые устройства в любом месте. На предприятиях все более интенсивно используются устройства разного типа, включая тонкие клиенты, смартфоны и планшеты. *Новое поколение сотрудников рассчитывает, что сможет использовать для работы свои собственные персональные устройства — как в офисе в рамках программы использования собственных устройств сотрудников (Bring-Your-Own-Device, BYOD), так и в более неформальном режиме в нерабочее время. Мобильным сотрудникам требуется возможность переходить с одного устройства (ноутбука, планшета или смартфона) на другое в течение дня.*

Пользователям, находящимся в удаленных подразделениях или филиалах, необходима гибкость в вопросах выбора подходящих устройств, соответствующих их потребностям, и возможность быстро менять устройства в случае возникновения технических проблем. ИТ-отделам требуется обеспечивать подрядчикам и партнерам доступ к приложениям и десктопам без необходимости предоставлять и обслуживать принадлежащие компании устройства.

Обеспечьте соответствие своей ИТ-стратегии требованиям завтрашнего дня. Сложно вводить изменения, если ваше внимание и ресурсы затрачиваются на поддержание традиционной инфраструктуры десктопов.

Благодаря этому вы можете более оперативно и эффективно реагировать на быстро меняющиеся требования к мобильности предприятий и гарантировать своевременность бизнес-преобразований в своей организации.

Пошаговая модель для мероприятия перехода на новое ПО**.** Как серьезное мероприятие, предполагающее вовлечение всей организации, проведение перехода от старого ПО к более поздней версии требует тщательно продуманной стратегии и детального планирования. Особенности такого подхода будут отличаться в каждой отдельной организации, но

обозначенные ниже шаги представляют собой универсальную модель, на которой должен строиться ваш процесс планирования.

Шаг 1: определение приоритетности бизнес-мероприятий

Проведение обновления является уникальным для каждой организации, которая внедряет решение. Выбор различных параметров, от способов доставки до конечных устройств (ПК, ноутбуков, тонких клиентов, планшетов), будет зависеть от таких факторов, как потребности различных типов пользователей внутри организации, задачи, которые они выполняют, и места, в которых они работают.

Шаг 2: сбор данных

Прежде чем приступить к анализу среды, важно собрать актуальные данные относительно пользователей, приложений и устройств, прибегнув к одному из трех подходов: сбору вручную, опросу или автоматизированному сбору.

Сбор данных вручную требует серьезных временных затрат, что делает его использование оправданным лишь для самых маленьких компаний.

Опрос экономит время работающей над проектом команды, но, как правило, приводит к нехватке данных, поскольку вероятность того, что каждый управляющий в организации заполнит опросник, невелика.

Недостаток *автоматизированного сбора данных* заключается в том, что его нельзя использовать для определения бизнес-характеристик, таких как критичность приложения или требования, касающиеся лицензирования и безопасности.

Максимально эффективный подход предполагает использование опроса для выявления бизнес-характеристик и автоматизированных инструментов для определения технических характеристик.

Шаг 3: анализ среды и сегментирование пользователей

Чтобы переход к более новой версии ПО и трансформация ИТ-отделов прошли максимально успешно, необходимо полное понимание имеющейся среды. Сюда относится следующее:

• *Сегментирование пользователей* по принципу привычных для них способов работы и общих требований.

• *Оценка существующих конечных устройств*, позволяющая установить, может ли использоваться устройство каждого типа после перехода к более поздней версии ОС и трансформации, и если да, то каким образом.

• *Анализ использующихся приложений*. Основываясь на таких факторах, как пользовательский профиль, требования в отношении ресурсов и технические требования, вы можете определить наилучший способ интеграции каждого.

Шаг 4: рационализация своих приложений

Опираясь на данные, полученные в шаге 2, можно сократить масштабы запланированного мероприятия, исключив необходимость выполнения лишней работы.

Шаг 5: решать связанные с совместимостью приложений вопросы в удобном темпе

Чтобы быстро приступить к переходу от старого ПО к более поздней версии, начать с приложений, про которые уже известно, что они будут совместимы с новой ОС, и приступить к реализации одной из множества стратегий по доставке приложений. Что касается приложений, которые не готовы к переходу на более позднюю версию ОС, то можно либо разрешить связанные с ними проблемы, основываясь на полученной информации, либо отметить их как подлежащие замене.

Заключение:

Сформированная таким образом среда позволяет добиться гибкости в вопросах предоставления приложений и десктопов по запросу, максимально соответствует потребностям предприятий и пользователей и обеспечивает необходимую современным сотрудникам мобильность, предоставляя возможность работать в любое время, в любом месте и на любом устройстве.

2.5 Создание в системе виртуальной машины для исполнения приложений

Технологии виртуализации

Согласно статистике средний уровень загрузки процессорных мощностей у серверов под управлением Windows не превышает 10%, у Unix-систем этот показатель лучше, но тем не менее в среднем не превышает 20%. Низкая эффективность использования серверов объясняется широко применяемым с начала 90-х годов подходом "одно приложение — один сервер", т. е. каждый раз для развертывания нового приложения компания приобретает новый сервер. Очевидно, что на практике это означает быстрое увеличение серверного парка и как следствие — возрастание затрат на его администрирование, энергопотребление и охлаждение, а также потребность в дополнительных помещениях для установки всё новых серверов и приобретении лицензий на серверную ОС.

Виртуализация ресурсов физического сервера позволяет гибко распределять их между приложениями, каждое из которых при этом "видит" только предназначенные ему ресурсы и "считает", что ему выделен отдельный сервер, т. е. в данном случае реализуется подход "один сервер — несколько приложений", но без снижения производительности, доступности и безопасности серверных приложений. Кроме того, решения виртуализации дают возможность запускать в разделах разные ОС с помощью эмуляции их системных вызовов к аппаратным ресурсам сервера.

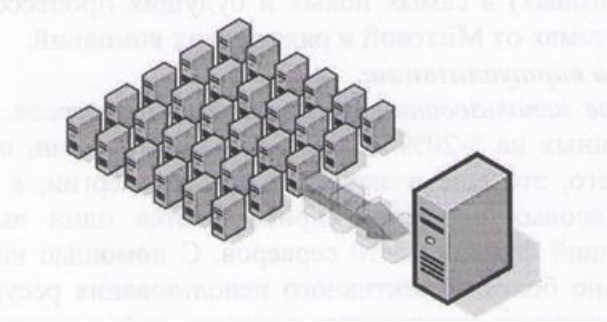


Рисунок 25 – Виртуализация подразумевает запуск на одном физическом компьютере нескольких виртуальных компьютеров

В основе виртуализации лежит возможность одного компьютера выполнять работу нескольких компьютеров благодаря распределению его ресурсов по нескольким средам. С помощью виртуальных серверов и виртуальных настольных компьютеров можно разместить несколько ОС и несколько приложений в едином местоположении. Таким образом, физические и географические ограничения перестают иметь какое-либо значение. Помимо энергосбережения и сокращения расходов благодаря более эффективному использованию аппаратных ресурсов, виртуальная инфраструктура обеспечивает высокий уровень доступности ресурсов, усовершенствованную систему восстановления в критических ситуациях.

В широком смысле понятие виртуализации представляет собой сокрытие настоящей реализации какого-либо процесса или объекта от истинного его представления для того, кто им пользуется. Продуктом виртуализации является нечто удобное для использования, на самом деле, имеющее более сложную или совсем иную структуру, отличную от той, которая воспринимается при работе с объектом. Иными словами, происходит отделение представления от реализации чего-либо. Виртуализация призвана абстрагировать программное обеспечение от аппаратной части.

В компьютерных технологиях под термином "виртуализация" обычно понимается абстракция вычислительных ресурсов и предоставление пользователю системы, которая "инкапсулирует" (скрывает в себе) собственную реализацию. Проще говоря, пользователь работает с удобным для себя представлением объекта, и для него не имеет значения, как объект устроен в действительности.

Сейчас возможность запуска нескольких виртуальных машин на одной физической вызывает большой интерес среди компьютерных специалистов, не только потому, что это повышает гибкость ИТ-инфраструктуры, но и потому, что виртуализация, на самом деле, позволяет экономить деньги.

Повышенный интерес к технологиям виртуализации в настоящее время неслучаен. Вычислительная мощь нынешних процессоров быстро растет, и вопрос даже не в том, на что эту мощь расходовать, а в том, что современная "мода" на двухъядерные и многоядерные системы, проникшая уже и в персональные компьютеры (ноутбуки и десктопы), как нельзя лучше позволяет реализовать богатейший потенциал идей виртуализации операционных систем и приложений, выводя удобство пользования компьютером на новый качественный уровень. Технологии виртуализации становятся одним из ключевых компонентов (в том числе, и маркетинговых) в самых новых и будущих процессорах Intel и AMD, в операционных системах от Microsoft и ряда других компаний.

Преимущества виртуализации:

1. *Эффективное использование вычислительных ресурсов:* Вместо 3х, а то 10 серверов, загруженных на 5-20% можно использовать один, используемый на 50- 70%. Кроме прочего, это еще и экономия

электроэнергии, а также значительное сокращение финансовых вложений: приобретается один высокотехнологичный сервер, выполняющий функции 5-10 серверов. С помощью виртуализации можно достичь значительно более эффективного использования ресурсов, поскольку она обеспечивает объединение стандартных ресурсов инфраструктуры в единый пул и преодолевает ограничения устаревшей модели "одно приложение на сервер".

2. *Сокращение расходов на инфраструктуру:* Виртуализация позволяет сократить количество серверов и связанного с ними ИТ-оборудования в информационном центре. В результате этого потребности в обслуживании, электропитании и охлаждении материальных ресурсов сокращаются, и на ИТ затрачивается гораздо меньше средств.

3. *Снижение затрат на программное обеспечение:* Некоторые производители программного обеспечения ввели отдельные схемы лицензирования специально для виртуальных сред. Так, например, покупая одну лицензию на Microsoft Windows Server 2008 Enterprise, вы получаете право одновременно её использовать на 1 физическом сервере и 4 виртуальных (в пределах одного сервера), а Windows Server 2008 Datacenter лицензируется только на количество процессоров и может использоваться одновременно на неограниченном количестве виртуальных серверов.

4. *Повышение гибкости и скорости реагирования системы:* Виртуализация предлагает новый метод управления ИТ-инфраструктурой и помогает ИТ- администраторам затрачивать меньше времени на выполнение повторяющихся заданий — например, на инициацию, настройку, отслеживание и техническое обслуживание. Многие системные администраторы испытывали неприятности, когда "рушится" сервер. И нельзя, вытащив жесткий диск, переставив его в другой сервер, запустить все как прежде... А установка? поиск драйверов, настройка, запуск... и на все нужны время и ресурсы. При использовании виртуального сервера – возможен моментальный запуск на любом "железе", а если нет подобного сервера, то можно скачать готовую виртуальную машину с установленным и настроенным сервером, из библиотек, поддерживаемых компаниями разработчиками гипервизоров (программ для виртуализации).

5. *Несовместимые приложения могут работать на одном компьютере:* При использовании виртуализации на одном сервере возможна установка linux и windows серверов, шлюзов, баз данных и прочих абсолютно несовместимых в рамках одной не виртуализированной системы приложений.

6. *Повышение доступности приложений и обеспечение непрерывности работы предприятия:* Благодаря надежной системе резервного копирования и миграции виртуальных сред целиком без перерывов в обслуживании вы сможете сократить периоды планового простоя и обеспечить быстрое восстановление системы в критических ситуациях. "Падение" одного виртуального сервера не ведет к потере остальных виртуальных серверов. Кроме того, в случае отказа одного физического сервера возможно произвести автоматическую замену на резервный сервер. Причем это происходит не заметно для пользователей без перезагузки. Тем самым

обеспечивается непрерывность бизнеса.

7. *Возможности легкой архивации:* Поскольку жесткий диск виртуальной машины обычно представляется в виде файла определенного формата, расположенный на каком-либо физическом носителе, виртуализация дает возможность простого копирования этого файла на резервный носитель как средство архивирования и резервного копирования всей виртуальной машины целиком. Возможность поднять из архива сервер полностью еще одна замечательная особенность. А можно поднять сервер из архива, не уничтожая текущий сервер и посмотреть положение дел за прошлый период.

8. *Повышение управляемости инфраструктуры:* использование централизованного управления виртуальной инфраструктурой позволяет сократить время на администрирование серверов, обеспечивает балансировку нагрузки и "живую" миграцию виртуальных машин.

Виртуальной машиной называется программная или аппаратная среда, которая скрывает настоящую реализацию какого-либо процесса или объекта от его видимого представления.

Виртуальная машина — это полностью изолированный программный контейнер, который работает с собственной ОС и приложениями, подобно физическому компьютеру. Виртуальная машина действует так же, как физический компьютер, и содержит собственные виртуальные (т.е. программные) ОЗУ, жесткий диск и сетевой адаптер.

ОС не может различить виртуальную и физическую машины. То же самое можно сказать о приложениях и других компьютерах в сети. Даже сама виртуальная машина считает себя "настоящим" компьютером. Но несмотря на это виртуальные машины состоят исключительно из программных компонентов и не включают оборудование. Это дает им ряд уникальных преимуществ над физическим оборудованием.

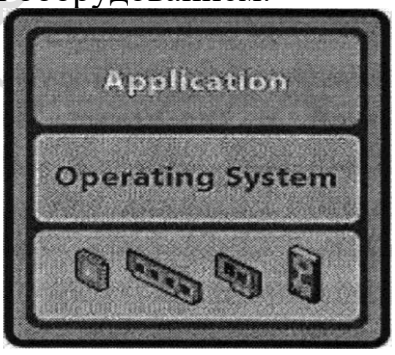


Рисунок 25 - Виртуальная машина

Основные особенности виртуальных машин более детально:

1. *Совместимость.* Виртуальные машины, как правило, совместимы со всеми стандартными компьютерами. Как и физический компьютер, виртуальная машина работает под управлением собственной гостевой операционной системы и выполняет собственные приложения. Она также содержит все компоненты, стандартные для физического компьютера (материнскую плату, видеокарту, сетевой контроллер и т.д.). Поэтому виртуальные машины полностью совместимы со всеми стандартными операционными системами, приложениями и драйверами устройств.

Виртуальную машину можно использовать для выполнения любого программного обеспечения, пригодного для соответствующего физического компьютера.

2. *Изолированность.* Виртуальные машины полностью изолированы друг от друга, как если бы они были физическими компьютерами Виртуальные машины могут использовать общие физические ресурсы одного компьютера и при этом оставаться полностью изолированными друг от друга, как если бы они были отдельными физическими машинами. Например, если на одном физическом сервере запущено четыре виртуальных машины, и одна из них дает сбой, это не влияет на доступность оставшихся трех машин. Изолированность — важная причина гораздо более высокой доступности и безопасности приложений, выполняемых в виртуальной среде, по сравнению с приложениями, выполняемыми в стандартной, невиртуализированной системе.

3. *Инкапсуляция.* Виртуальные машины полностью инкапсулируют вычислительную среду. Виртуальная машина представляет собой программный контейнер, связывающий, или "инкапсулирующий" полный комплект виртуальных аппаратных ресурсов, а также ОС и все её приложения в программном пакете. Благодаря инкапсуляции виртуальные машины становятся невероятно мобильными и удобными в управлении. Например, виртуальную машину можно переместить или скопировать из одного местоположения в другое так же, как любой другой программный файл. Кроме того, виртуальную машину можно сохранить на любом стандартном носителе данных: от компактной карты Flash-памяти USB до корпоративных сетей хранения данных.

4. *Независимость от оборудования.* Виртуальные машины полностью независимы от базового физического оборудования, на котором они работают. Например, для виртуальной машины с виртуальными компонентами (ЦП, сетевой картой, контроллером SCSI) можно задать настройки, абсолютно не совпадающие с физическими характеристиками базового аппаратного обеспечения. Виртуальные машины могут даже выполнять разные операционные системы (Windows, Linux и др.) на одном и том же физическом сервере. В сочетании со свойствами инкапсуляции и совместимости, аппаратная независимость обеспечивает возможность свободно перемещать виртуальные машины с одного компьютера на базе x86 на другой, не меняя драйверы устройств, ОС или приложения. Независимость от оборудования также дает возможность запускать в сочетании абсолютно разные ОС и приложения на одном физическом компьютере.

Рассмотрим основные разновидности виртуализации, такие как:

– виртуализация серверов (полная виртуализация и паравиртуализация);

– виртуализация на уровне операционных систем;

– виртуализация приложений;

– виртуализация представлений.

Паравиртуализация (paravirtualization). Модификация ядра гостевой

ОС выполняется таким образом, что в нее включается новый набор API, через который она может напрямую работать с аппаратурой, не конфликтуя с другими виртуальными машинами. При этом нет необходимости задействовать полноценную ОС в качестве хостового ПО, функции которого в данном случае исполняет специальная система, получившая название гипервизора (hypervisor). Именно этот вариант является сегодня наиболее актуальным направлением развития серверных технологий виртуализации и применяется в VMware ESX Server, Хеп (и решениях других поставщиков на базе этой технологии), Microsoft Hyper-V.

Виртуализация на уровне ядра ОС (operating system-level virtualization). Этот вариант подразумевает использование одного ядра хостовой ОС для создания независимых параллельно работающих операционных сред. Для гостевого ПО создается только собственное сетевое и аппаратное окружение. Такой вариант используется в Virtuozzo (для Linux и Windows), OpenVZ (бесплатный вариант Virtuozzo) и Solaris Containers. Достоинства — высокая эффективность использования аппаратных ресурсов, низкие накладные технические расходы, отличная управляемость, минимизация расходов на приобретение лицензий. Недостатки — реализация только однородных вычислительных сред.

Виртуализация приложений подразумевает применение модели сильной изоляции прикладных программ с управляемым взаимодействием с ОС, при которой виртуализируется каждый экземпляр приложений, все его основные компоненты: файлы (включая системные), реестр, шрифты, INI-файлы, COM-объекты, службы. Приложение исполняется без процедуры инсталляции в традиционном ее понимании и может запускаться прямо с внешних носителей (например, с флэш-карт или из сетевых папок).

Виртуализация представлений (рабочих мест) Виртуализация представлений подразумевает эмуляцию интерфейса пользователя. Т.е. пользователь видит приложение и работает с ним на своем терминале, хотя на самом деле приложение выполняется на удаленном сервере, а пользователю передается лишь картинка удаленного приложения. В зависимости от режима работы пользователь может видеть удаленный рабочий стол и запущенное на нем приложение, либо только само окно приложения.

2.6 Решение проблем конфигурации с помощью групповых политик

Групповая политика – это один из главных инструментов для настройки Windows, с его помощью можно как задействовать множество функций системы, так и отключить их. С помощью групповых политик можно настроить систему для пользователя под которым вы вошли в систему, для других пользователей данного компьютера или для других пользователей на других компьютерах в доменной сети.

Групповые политики доступны только в профессиональных, максимальных и корпоративных версиях Windows, у пользователей домашней Windows данный набор инструментов отключен.

Централизованные групповые политики. Зачастую во множестве организаций используют централизованное управление групповыми политиками. Например, в офисе около двадцати компьютеров, есть один сервер и между этими компьютерами настроена одна закрытая сеть, к которой может подключить только администратор под своим именем и паролем. В этой закрытой сети на контролере домена в Active Directory можно задавать различные доступы, закрывать или открывать различные ресурсы. Также возможно задать групповые политики для всех ПК в сети: установить открытие определенных сайтов в качестве домашней страницы, запретить доступ к различным настройкам панели управления, закрыть доступ к определенным папкам, запретить смену заставки и т.д.

В групповых политиках есть множество инструментов, чтобы настроить систему под нужные предпочтения, там есть практически всё, что видит перед собой пользователь на компьютере, и все это можно настраивать. Системный администратор может менять настройки компьютера с помощью групповых политик на сервере.

Локальные групповые политики. Групповые политики используются не только системными администраторами, их можно использовать для настройки локального компьютера (если не домашняя Windows).

К примеру, с помощью групповых политик можно изменить экран входа в Windows 7, также можно ограничить доступ к некоторым папкам другим пользователям данного компьютера, можно управлять паролями, отображением папок, отключить автоматическое обновление, разрешать запуск только определенных приложений, отключить доступ к панели управления и т.д.

Можно добавить через консоль групповые политики и настроить систему для других пользователей данного компьютера.

Чтобы открыть групповые политики нужно:

1.Чтобы была Windows профессиональная, корпоративная или максимальная;

2. В строке поиска или в меню выполнить (выполнить вызывается клавишами Win+R) написать команду **gpedit.msc** и нажать клавишу Enter.

3. В открывшемся редакторе локальной групповой политики найти нужные параметры, отключить или включить их. К примеру, можно отключить автозапуск съемных носителей.

В левой колонке открыть “Конфигурация компьютера” => Административные шаблоны => Компоненты Windows => Политики автозапуска => с правой стороны открыть политику “Отключить автозапуск”.

Поставить точку возле “Включить” => нажать “Ок”.

После каждого изменения групповых политик нужно перезагружать компьютер, чтобы изменения вступили в силу

Работа в редакторе. Разделяется главное окно управления на две части. Слева располагается структурированные категории политик. Они в свою

очередь делятся еще на две различные группы – настройка компьютера и настройка пользователя.

В правой части отображается информация о выбранной политике из меню слева.

Из этого можно сделать вывод, что работа в редакторе осуществляется путем перемещения по категориям для поиска необходимой настройки. Выбрать, например, *«Административные шаблоны»* в *«Конфигурации пользователя»* и перейдите в папку *«Меню «Пуск» и диспетчер задач»*. Теперь справа отобразятся параметры и их состояния. Нажать на любую строку, чтобы открыть ее описание.

Настройки политики. Каждая политика доступна для настройки. Открывается окно редактирования параметров по двойному щелчку на определенную строку. Вид окон может отличаться, все зависит от выбранной политики.

Стандартное простое окно имеет три различных состояния, которые настраиваются пользователем. Если точка стоит напротив *«Не задано»*, то политика не действует. *«Включить»* – она будет работать и активируются настройки. *«Отключить»* – находится в рабочем состоянии, однако параметры не применяются.

Рекомендуется обратить внимание на строку *«Поддерживается»* в окне, она показывает, на какие версии Windows распространяется политика.

Фильтры политик. Минусом редактора является отсутствие функции поиска. Существует множество различных настроек и параметров, их больше трех тысяч, все они разбросаны по отдельным папкам, а поиск приходится осуществлять вручную. Однако данный процесс упрощается благодаря структурированной группе из двух ветвей, в которых расположились тематические папки.

Например, в разделе *«Административные шаблоны»*, в любой конфигурации, находятся политики, которые никак не связаны с безопасностью. В этой папке находится еще несколько папок с определенными настройками, однако можно включить полное отображение всех параметров, для этого нужно нажать на ветвь и выбрать пункт в правой части редактора *«Все параметры»*, что приведет к открытию всех политик данной ветви.

Экспорт списка политик**.** Если все-таки появляется необходимость найти определенный параметр, то сделать это можно только путем экспорта списка в текстовый формат, а потом уже через, например Word, осуществлять поиск. В главном окне редактора есть специальная функция *«Экспорт списка»*, он переносит все политики в формат TXT и сохраняет в выбранном месте на компьютере.

Применение фильтрации**.** Благодаря появлению ветви *«Все параметры»* и улучшению функции фильтрации поиск практически не нужен, ведь лишнее откидывается путем применения фильтров, а отображаться будут только необходимые политики.

Процесс применения фильтрации:

1. Выбрать, например, *«Конфигурация компьютера»*, открыть раздел *«Административные шаблоны»* и перейти в *«Все параметры»*.

2. Развернуть всплывающее меню *«Действие»* и перейти в *«Параметры фильтра»*.

3. Поставить галочку возле пункта *«Включить фильтры по ключевым словам»*. Здесь имеется несколько вариантов подбора соответствий. Открыть всплывающее меню напротив строки ввода текста и выбрать *«Любой»* – если нужно отображать все политики, которые соответствуют хотя бы одному указанному слову, *«Все»* – отобразит политики, содержащие текст из строки в любом порядке, *«Точный»* – только параметры, точно соответствующие заданному фильтру по словам, в правильном порядке. Флажками снизу строки соответствий отмечаются места, где будет осуществляться выборка.

Нажать *«ОК»* и после этого в строке *«Состояние»* отобразятся только подходящие параметры.

В том же всплывающем меню *«Действие»* ставится или убирается галочка напротив строки *«Фильтр»*, если нужно применить или отменить заранее заданные настройки подбора соответствий.

Некоторые настройки групповой политики:

1. Отслеживать вход в аккаунт

С помощью групповой политики можно заставить Windows записывать все успешные и неудачные попытки входа на ПК с любой учетной записи. Можно использовать такую информацию, чтобы отслеживать, кто входит в систему на ПК и пытался ли кто-то войти в систему или нет.

В редакторе групповой политики перейти в *Конфигурация компьютера → Конфигурация Windows → Параметры безопасности → Локальные политики* → Политика аудита и дважды щелкнуть Аудит событий входа в систему.

Для просмотра этих журналов потребуется доступ к другому инструменту Windows – Windows Event Viewer. Снова открыть диалоговое окно «Выполнить» и введите в нем **eventvwr**, чтобы открыть средство просмотра событий Windows. Здесь развернуть «Журналы Windows», а затем выбрать опцию «Безопасность». На средней панели просмотреть все недавние события. Успешные события входа в систему имеют идентификатор события: 4624, а неудачные – идентификатор события: 4625. Найти эти идентификаторы событий, чтобы найти логины и увидеть точную дату и время входа. Двойной щелчок по этим событиям покажет более подробную информацию вместе с точным именем учетной записи пользователя, который вошел в систему.

2. Запретить доступ к Панели управления. *Конфигурация пользователя → Административные шаблоны* → Панель управления

3. Запретить пользователям устанавливать программы. *Конфигурация компьютера → Административные шаблоны → Компоненты Windows* → Установщик Windows

4. Отключите съемные запоминающие устройства. *Конфигурация пользователя → Административные шаблоны → Система* → Доступ к съемным носителям

5. Запретить запуск определенных приложений. *Конфигурация пользователя → Административные шаблоны →* Система

6. Отключите командную строку и редактор реестра. *Конфигурация пользователя → Административные шаблоны →* Система

7. Скрыть диски с разделами. *Конфигурация пользователя → Административные шаблоны → Компоненты Windows →* Проводник

8. Настройки для меню «Пуск» и панели задач. *Конфигурация пользователя → Административные шаблоны →* Меню «Пуск» и панель задач

2.7 Тестирование на совместимость в безопасном режиме. Восстановление системы

Безопасный режим — это режим запуска системы Windows, предназначенный для устранения неполадок, в котором используется ограниченный набор служб и компонентов. Загружаются только базовые файлы и драйверы, необходимые для запуска Windows. В углах экрана отображаются слова Безопасный режим, соответствующие используемому режиму Windows.

Если при загрузке в безопасном режиме существующие проблемы не возникают, то из списка возможных причин можно исключить параметры, используемые по умолчанию, и базовый набор драйверов устройств. Если причина проблемы неизвестна, можно воспользоваться методом исключения, чтобы ее обнаружить. Необходимо поочередно осуществить запуск всех обычно используемых программ, включая программы из папки «Автозагрузка», чтобы увидеть, какая из программ может приводить к возникновению проблемы.

Если компьютер автоматически, без запроса, запускается в безопасном режиме, возможно, проблема препятствует обычной загрузке Windows.

Со второй половины 2017 года компания Майкрософт официально прекратила поддержку седьмой версии Windows. В связи с этим, многие пользователи стараются найти варианты для самостоятельного решения многих ошибок и проблем.

Согласно последним данным компании, обновления для ОС выходят намного реже, а поддержка пользователей оказывается только в действительно важных случаях. Весь упор корпорация делает на усовершенствование Windows 10. Недостаточная поддержка системы приводит к возникновению багов и зависаний.

Часто, действия пользователей также являются причиной слишком медленной работы ОС. Установка «тяжелых» игр, программ, перезагрузка оперативной памяти и жесткого диска – это основные факторы поломки системы. Если вы столкнулись с резким ухудшением работы компьютера, следует провести восстановление. После этой процедуры, все функции ПК будут исправлены.

Существует три базовых метода восстановления:

– с помощью встроенного помощника;

– с использованием системных инструментов (BIOS, командная строка и прочие);

– через сторонние программы.

Стандартная утилита «Восстановление»

По умолчанию, каждая копия ОС Windows 7 периодически создает точки восстановления – это архивированная версия последней успешной конфигурации ПК, которую пользователь может использовать для восстановления.

Каждая такая точка восстановления хранится на жестком диске компьютера. Чтобы выбрать одну из них, необходимо воспользоваться стандартной утилитой Windows. Этот вариант является самым простым и подойдет только в том случае, если операционная система нормально загружается и на компьютере не отключена функция создания архивированных копий ОС.

Порядок действий:

1. Зайти в панель управления ПК и в текстовом поле для поиска ввести «Восстановление системы»;

2. Во вкладке результатов выбрать одноименное окно и дождаться его открытия;

3. Нажав кнопку «Далее», дается согласие на обработку личных данных системой. В процессе восстановления будут сохранены только те файлы и папки, которые были созданы до даты добавления выбранной точки доступа. Изменяются также и настройки конфигурации оперативной памяти и жесткого диска. Также, будут обновлены драйвера и другое ПО, которое поддерживает стабильную работу системы. Компьютер должен быть подключен к интернету;

4. Выбирая точку восстановления, обратить внимание на дату ее создания. Дата сохранения должна соответствовать тому периоду, когда Windows 7 работала в нормальном режиме, без сбоев;

5. Нажать на галочку «Показать другие точки», чтобы увидеть все объекты окна. Чтобы посмотреть с какими приложениями будет работать процесс отката, выбрать необходимую точку и нажать на клавишу «Затрагиваемые программы»;

6. Выбрать созданную резервную копию, нажать «Далее»;

7. В новом окне подтвердить свой выбор. Проверить свойства точки и наименование диска, с которым она будет работать (для восстановления ОС в соответствующей графе должен быть указан системный диск С);

Далее утилита начнет свою работу. Это займет не более 30 минут. После перезагрузки компьютера все ошибки будут устранены, а система заработает в нормальном режиме.

Использование безопасного режима

После возникновения серьезных ошибок в работе Windows 7, система может не загружаться в обычном режиме. Для устранения такой неполадки предусмотрена возможность запуска в безопасном режиме. С его помощью можно загрузить Windows специально для устранения неполадок.

В этом варианте загрузки отсутствуют некоторые стандартные службы и параметры. Выполняется запуск только базовых драйверов и компонентов, которые необходимы для работы системы. Такой вариант подойдет в том случае, если вы не можете работать из-за постоянно возникающей ошибки, которая завершает работу компьютера или вызывает его зависание.

В безопасном режиме подобные ошибки не будут появляться, и можно удалить вредоносные программы или выполнить откат системы.

Чтобы запустить безопасный режим:

- открыть меню запуска, удерживая клавишу F8, F12 или Escape, в зависимости от модели ПК;
- через несколько секунд появиться окно параметров загрузки ОС Windows 7;
- выбрать пункт «Безопасный режим»,
- нажмите Enter для выполнения действия.

После перезагрузки ПК появится рабочий стол Windows 7, где можно выполнить восстановление через точку доступа.

Примечание! *В этом режиме нет возможности использовать соединение с глобальной сетью. Если для настройки нормальной работы системы потребуется доступ к интернету, в окне дополнительных параметров выбрать пункт «Безопасный режим с поддержкой сетевых компонентов».*

Автоматическое устранение неполадок

С помощью параметров загрузки также можно включить опцию устранения неполадок. Она позволяет выполнить сброс всех ошибок в автоматическом режиме. Также, у пользователей есть возможность включить последнюю нормальную конфигурацию Windows 7. После выбора этой функции, рабочий стол ОС загрузится в виде последней удачной копии параметров.

Порядок действий:

- на этапе включения компьютера запустить системное меню параметров включения, удерживая F8;
- выберать пункт «Последняя удачная загрузка» и попытаться выполнить запуск Windows 7;
- в случае неудачной попытки включения, снова вернуться в системное меню и выбрать пункт «Устранения неполадок»;
- операционная система запустит утилиту для автоматического исправления всех ошибок. После этого, Windows 7 должна запуститься в уже восстановленном виде.

Восстановление через командную строку

Этот вариант подойдет в том случае, если на компьютере нет ни одной сохраненной точки восстановления, но удается запустить Windows 7.

Выполнить включение ПК в безопасном режиме с поддержкой командной строки. Это позволит открыть окно CMD и работать с базовыми командами.

После включения Windows 7, открыть окно «Выполнить» и введите в поле команду **cmd**. Она позволит запустить строку.

В открывшемся окне напечатать команду **rstrui.exe** и нажать Ввод. Через несколько минут появиться сообщение об успешном восстановлении ПК. Перезагрузить его, чтобы выйти из безопасного режима.

Восстановление с помощью флешки или диска

Из-за возникновения фатальных ошибок в Windows 7, пользователям не удается выполнить запуск системы с помощью безопасного режима. В таком случае, восстановить Windows можно используя загрузочный диск или флешку.

Загрузочный носитель – это съемное устройство (чаще CD или flash-накопитель), на которое скачивается установочная копия операционной системы Windows. Создать такой носитель можно на другом ПК с помощью программ ISO Maker, Live CD, Daemon Tools и прочих утилит. Если есть диск с официальной копией Windows, его также можно использовать для отката системы.

Примечание! *Обратите внимание! Версии системы Windows 7 на загрузочном диске и на компьютере должны быть одинаковыми. В противном случае, восстановление выполнить не удастся.*

Перед началом исправления работы с помощью съемного носителя, следует настроить очередь загрузки компонентов в BIOS – компьютер должен загружать не установленную версию Windows 7, а ту, которая находится на накопителе:

- открыть BIOS, нажав на клавиши F8 или F12 сразу после включения ПК;
- перейдите во вкладку Boot;
- в открывшемся окне найти пункты «Boot Priority» — каждый из них означает определенный порядок загрузки компонентов ПК. На первое место выставить тип загрузочного носителя. Если восстановление будет проводиться с флешки, в первой графе «Boot Priority» выбрать «USB Storage». Если с диска, выставить на первое место компонент «Hard Drive»;
- теперь, в результате запуска компьютера, появится окно установки Windows Найти пункт «Восстановление системы» и нажать на него;
- подтвердить действие еще раз:
- нажать на клавишу «Далее» и выполнить выбор точки доступа.

*Возникновение ошибки 0*000000*

Иногда, во время восстановления или сразу после завершения операции может возникнуть ошибка с кодом 0*000000. Ее разные вариации могут дополняться другими идентификаторами, к примеру, 0*c0000034 или 0*0000007b.

После повторного включения система завершит восстановление и будет работать без сбоев.

Возникновение неполадки информирует пользователя о том, что системе не удалось получить доступ к процедуре чтения с загрузочного носителя. Так как ошибка появилась в уже установленной системе сразу после ее восстановления, можно говорить о том, что ее причина – это проблема с драйверами.

Скорее всего, после процедуры восстановления режим работы системного диска был изменен из IDE на AHCI, поэтому существующие драйверы не подошли для выполнения загрузки ОС. Решить проблему можно, отключив AHCI через BIOS:

- открыть меню BIOS;
- зайти во вкладку настроек CMOS и выбрать пункт конфигурации последовательного интерфейса SATA, как показано на рисунке ниже;

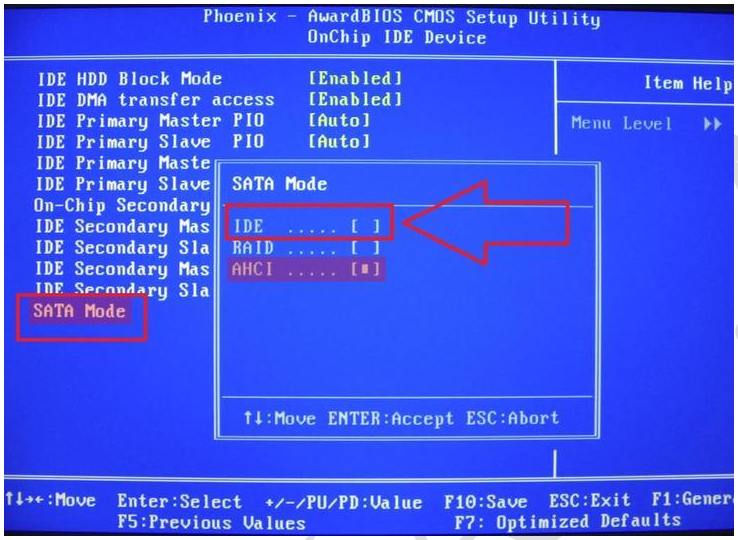


Рисунок 26 – Переключение режимов SATA в BIOS

- нажать Enter и во всплывающем окне выберать параметр IDE;
- сохраните настройки, выйти из BIOS и перезагрузить компьютер.

После повторного включения система завершит восстановление и будет работать без сбоев.

2.8 Производительность ПК. Проблемы производительности. Анализ журналов событий

Основные признаки медленной работы компьютера:

1. Долгое включение и такое же долгое выключение компьютера.
2. Многие программы очень долго запускают, загрузка игр занимает очень много времени.
3. Реакция курсора может не успевать за движениями мышки, притормаживать, или наблюдается поздняя реакция на клик.
4. И другие признаки замедленной работы ПК.

Причины торможения компьютера:

1. Плохая совместимость программного обеспечения, операционной системы, игр с конфигурацией АО самого компьютера.
2. Большая фрагментация файловой системы, на жестком диске скопилось много программного хлама.
3. Проблемы с операционной системой, сбои, неправильная настройка.

4. Большое количество запущенных служб, в которых нет необходимости.

5. Заражение компьютера вредоносным программным обеспечением.

6. Программные конфликты (между драйверами, антивирусами и др.).

7. Плохая терморегуляция, выход из строя кулера, засорение радиаторов.

8. Нарушение основных параметров в BIOS.

Существует множество способов, позволяющих ускорить работу компьютера:

1. Апгрейд (модернизация) ПК.

2. Чистка внутренностей, ремонт неисправных элементов системы охлаждения.

3. Очистка жесткого диска от всего ненужного, выполнить дефрагментацию.

4. Переустановить операционную систему.

5. Оптимизация работы ОС.

6. Настроить BIOS.

7. Отключить все лишнее, контролировать автозагрузку.

8. Выполнить обновление драйверов.

9. Установить операционную систему соответствующую ПК.

10. Проверить систему на вирусы.

Определение производительности

В операционной системе Windows 7 уже встроена специальная функция для проверки производительности системы и быстродействия компьютера.

Открыть меню «Пуск» и в специальном поисковом поле (оно находится внизу меню) ввести «произ». Среди результатов поиска будет программа «Повышение производительности компьютера». Запустить ее.

Снизу сделать клик по кнопке «Повторить оценку». Программа начнет анализировать продуктивность системы и по завершению выдаст результаты, по которым можно будет сделать вывод о быстродействии компьютера.

После этого можно приступать к повышению производительности компьютера выбранными способами.

Апгрейд комплектующих компьютера

В большинстве случаев достаточно заменить только один или несколько слабых элементов, чтобы вся система начала быстрее работать:

- *Центральный процессор*. Смысл замены будет действительно оправдан, только если новый намного превзойдет старого по мощности, не менее чем на 30%.
- *Оперативная память*. Ее никогда не бывает много, можно смело увеличивать оперативную память. Когда компьютер сильно загружен, открыть диспетчер задач на вкладке быстродействия и посмотрите уровень используемой памяти. Если он превышает 80%, то можно увеличивать память в полтора два раза.
- *Винчестер*. И суть тут вовсе не в его объеме и количестве свободного места. Основная суть быстродействия жесткого диска заключается в

скорости вращения его мотора. К примеру, если имеется диск со скоростью 5400 оборотов и заменить него на диск со скоростью 7200 оборотов. Это намного более ощутимо добавит скорости для систем, особенно скорости записи/чтения.

• *Видеокарта*. Большинству понятно, что для более реалистичной графики необходимо иметь мощный видеоадаптер. Так что, если видеокарта не имеет хорошей производительности, необходимо менять на новую, которая гораздо превосходит ее по мощности. Главное – нужно учитывать, что для мощной видеокарты нужен и мощный процессор.

Чтобы приблизительно определить слабые места системы, воспользуйтесь встроенной функцией оценки производительности компьютера. Определить «слабое звено» можно по оценкам, которые выводятся после проверки производительности. Там, где самая низкая оценка, в том направлении и необходимо увеличивать мощность ПК.

Чистка внутренностей, ремонт неисправных элементов системы охлаждения

Различные неисправности в системе охлаждения также способны существенно замедлить работу всей системы. Если ломается кулер на центральном процессоре, то это привод к его перегреву и снижению тактовой частоты.

Перегрев может возникать даже при исправной системе охлаждения. Снять крышку системного блока, и очень аккуратно почистить системный блок, чем можно увеличить скорость работы компьютера, а также продлить срок службы его компонентов.

Очистка жесткого диска от всего ненужного, выполнение дефрагментации

Самое первое что нужно делать для повышения производительности системы. Дефрагментация позволяет собирать различные фрагменты программ, которые находятся в разных частях жесткого диска, в одно место на диске. Благодаря этому, считывающему устройству винчестера не нужно выполнять много лишних перемещений по дискам, ведь все находится в одном месте. Таким образом и повышается производительность.

Кроме того, нужно избавиться от лишней информации и всякого программного хлама, который накапливается на диске со временем работы. Особенно это важно, когда в разделе операционной системы практически нет свободного места. Если места менее 2 Гб, система теряет свою производительность. Необходимо следить, чтобы на нем было процентов тридцать свободного места, если это конечно возможно.

Переустановка операционной системы

Этот шаг почти всегда помогает увеличить скорость работы компьютера. В некоторых случаях, производительность может увеличиться в три раза. Просто такова суть операционной системы, со временем в ней накапливаются различные ошибки, она забивается ненужными службами, которые даже выполняют серьезные изменения в самой системе. Это и многое другое приводит к ухудшению скорости работы компьютера, на многие операции теперь требуется гораздо больше времени.

Если тщательно следить за чистотой системы и ничего туда не устанавливать, то можно годами пользоваться одной и той же Windows. Но чаще всего, на компьютере постоянно происходит движение: устанавливаются и удаляются программы, обновляются драйвера, загружаются большие объемы различной информации — в таких условиях система постепенно начинает работать медленнее. Лучше всего, для профилактики, где-то раз в год форматировать диск и с чистого листа устанавливать новую операционную систему.

Оптимизация работы ОС

Программа PCMedic. Главная особенность этой утилиты, это полная автоматизация всех операций. Нужно только выбрать подходящие параметры и запустить процесс настройки.

Программа состоит только из одного главного окна. Здесь можно выбрать установленную операционную систему, тип центрального процессора (например, Intel или AMD), дальше нужно выбрать один из двух способов оптимизации — Heal (очистка системы), либо Heal & Boost (кроме очистки выполняется еще и ускорение). После того, как выбираются все параметры нажать на кнопку «Go» – программа выполнит все необходимые изменения.

Программа Ausloqics BoostSpeed. Она состоит из нескольких утилит, позволяющих выполнять оптимизацию системы практически во всех направлениях. Используя это приложение можно провести дефрагментацию, почистить файловую систему, очистить реестр, увеличить скорость работы интернета и еще много другого. Программа обладает встроенным советчиком, который помогает определить приоритетные направления в оптимизации системы. Хотя следует внимательно смотреть, действительно ли эти все действия необходимы.

Программа Ccleaner. Способна почистить диск от ненужных, временных фалов и выполнить очистку реестра. Благодаря удалению ненужных файлов, можно увеличить количество свободного места на жестком диске. А вот при чистке реестра, особого повышения производительности не наблюдается. Зато если будет случайно удален какой-нибудь важный параметр, система начнет выдавать ошибки и это может привести к серьезным сбоям.

ВНИМАНИЕ! Перед тем, как выполнять все эти действия, настоятельно рекомендуется создать точку восстановления!

ВСЕГДА смотрите файлы, которые удаляют утилиты для очистки системы. Бывают случаи безвозвратного удаления нужных и даже важных файлов, которые программы ложно приняли за ненужные, или временные файлы.

Настройка BIOS

В БИОСе хранятся параметры компьютера, отвечающие за оборудование, загрузку ОС, время и других ключевые элементы. Чаще всего настройки БИОСа не вызывают никакого снижения производительности. Но в редких случаях, при неправильных критических параметрах, компьютер может начать тормозить.

Если есть сомнения в правильности настроек, то можно воспользоваться опцией автоматической настройки оптимальных параметров «Load Optimal Settings» (название функции может быть иным, в зависимости от производителя). После этого сохранить все настройки и выполнить перезагрузку компьютера.

Отключение всего лишнего, контроль автозагрузки

Практически каждая программа пытается прописать себя в автозагрузку. Постепенно, программы в автозагрузке накапливаются и их всех надо запускать, при каждом старте системы. Из-за этого компьютер очень долго включается и выключается. Дополнительно, после автозагрузки, все эти приложения остаются в рабочем состоянии, используя ресурсы. Поэтому, лучше всего на панели задач удалять ненужные приложения, или хотя бы отключать для них автозагрузку.

Что бы посмотреть все приложения, которые стартуют вместе с Windows, и отключить лишние, одновременно зажать две клавиши Win+R и в следующем окне написать msconfig, затем нажать Enter. Появится оконо конфигурации системы, теперь перейти в раздел автозагрузки. Здесь будет находиться список приложений, стоящих в автозагрузке. Просто снять флажки со всех, которыми не используются. В случае чего, программу всегда можно вернуть в автозагрузку, установив нужный флажок. Главное иметь представление, что это за программы и какое их назначение.

Обновление драйверов

Этот шаг может дать нужных эффект, если установлены устаревшие драйвера или те, что установились вместе с системой. Больше всего может влиять драйвер для материнской платы, хотя и другие неплохо воздействуют на скорость работы.

Надежнее всего вручную выполнять обновление драйверов. Но можно воспользоваться специальными утилитами, которые способны сами определять необходимые драйвера, находить их в интернете и устанавливать. Например, можно воспользоваться Drover Checker.

Установка операционной системы соответствующей ПК

Если, например, компьютер, с 2 Гб оперативной памяти и используется Windows XP, то стоит задуматься о переходе на Windows 7, после этого сразу почувствуется положительная разница. А если на компьютере двухъядерный процессор и 4 Гб памяти (а может даже больше), то необходимо использовать 64х разрядную Windows 7.

Чистка системы от вирусов

Это одна из самых распространенных причин медленной работы компьютера. Если стало заметно, что компьютер внезапно начал долго выполнять команды, срочно выполнить сканирование на наличие вредоносного ПО. Лучше всего выполнить сканирование системы штатным антивирусом (например, Антивирусом Касперского), а затем просканировать дополнительными антивирусными утилитами, типа Dr. Web CureIt, AVZ и др.

Контрольные вопросы:

1. Анализ производительности ПК. Проблемы производительности. Анализ журналов событий

2. Проблемы перехода на новые версии программ

3. Анализ приложений с проблемами совместимости

4. Решение проблем конфигурации с помощью групповых политик

5. Тестирование на совместимость в безопасном режиме. Восстановление системы

6. Понятие совместимости программного обеспечения

Заключение

Предлагаемый курс лекций по МДК.04.01 Внедрение и поддержка компьютерных систем содержит теоретический материал с вопросами.

Весь материал структурирован в соответствии с рабочей программой и включает в себя теоретический материал по разделам и темам, необходимые примеры, иллюстрации, вопросы.

Результатом изучения материала являются полученные знания об основных методах и средствах эффективного анализа функционирования программного обеспечения, а также, о загрузке и установке программного обеспечения.

Список использованных источников

1 Кугаевских, А.В. Проектирование информационных систем. Системная и бизнес-аналитика : учебное пособие : [16+] / А.В. Кугаевских ; Новосибирский государственный технический университет. – Новосибирск : Новосибирский государственный технический университет, 2018. – 256 с. : табл., схем., ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=573827

2 Ипатова, Э.Р. Методологии и технологии системного проектирования информационных систем : учебник / Э.Р. Ипатова, Ю.В. Ипатов. – 2-е изд., стер. – Москва : ФЛИНТА, 2016. – 257 с. : табл., схем. – (Информационные технологии). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=79551

3 https://standartgost.ru/0/757-programmnoe_obespechenie