

Министерство образования и науки Российской Федерации
Байкальский государственный университет экономики и права

В.В.Братищенко

ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

Иркутск

Издательство БГУЭП

2004

В.В.Братищенко

ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

УДК 004.82 (075.8)
ББК 32.97я7

Б.87

Печатается по решению редакционно-издательского совета
Байкальского государственного университета экономики и права

Рецензенты канд. физ.-мат. наук, доц. В.В.Ступин
 канд. физ.-мат. наук, доц. С.С.Сосинская

Братищенко В.В.
Б.87 Проектирование информационных систем: Учеб.пособие. —
Иркутск: Изд-во БГУЭП, 2004.— 84с.
ISBN 5-7253-1066-3

Содержит сведения о стадиях и этапах проектирования различных моделей жизненного цикла информационной системы; описание последовательности и методологии выполнения работ на разных этапах проектирования; методы организации работ по проектированию.

Предназначено для студентов специальности 351400 — Прикладная информатика в экономике.

ББК 32.97я7

ISBN 5-7253-1066-3

© Братищенко В.В., 2004
© Издательство БГУЭП, 2004

ОГЛАВЛЕНИЕ

1. Определение, компоненты и классификация информационных систем	4
2. Жизненный цикл информационной системы	11
3. Принципы разработки информационных систем	20
4. Методология разработки	24
5. Планирование развития информационной системы	28
5.1. Изучение и анализ организации	29
5.2. Реинжиниринг бизнес – процессов	32
5.3. Планирование архитектуры информационной системы	33
6. Стадия системного анализа	34
6.1. Описание и анализ существующей технологии управления	34
6.2. Определение требований и приоритетов	40
6.3. Содержание технического задания	40
7. Конструирование	45
7.1. Работы стадии конструирования	45
7.2. CASE-технология проектирования информационных систем	47
7.3. Построение функциональной модели информационной системы ..	50
7.4. Построение модели данных	54
7.4.1. Структурированное описание данных	54
7.4.2. Модели данных для СУБД	56
7.4.3. Инфологическое проектирование. Модель «Сущность-связь»	57
7.4.4. Анализ событий (Event Analysis)	61
7.4.5. Даталогическое проектирование	62
7.4.6. Иерархическая модель	63
7.4.7. Сетевая модель	64
7.4.8. Реляционная модель	65
7.5. Определение набора приложений. Проектирование интерфейса пользователя, структуры и логики работы программы	67
8. Реализация и сопровождение	69
9. Организация процесса разработки информационной системы	71
9.1. Управление основными характеристиками проекта	71
9.2. Организация коллективной работы над проектом	72
9.3. Экономические показатели проектирования	74
ПРИЛОЖЕНИЕ Требования к содержанию проектных документов	76
Список использованной литературы	84

1. Определение, компоненты и классификация информационных систем

Термин "Информационная система" появился вместе с техническими средствами обработки информации, когда возникла необходимость в описании процесса обработки. В современном понимании *информационная система (ИС)* - это организационные, технические, программные и информационные средства, объединенные в единую систему с целью сбора, хранения, обработки и выдачи информации, предназначенной для решения определенного множества задач.

Существует руководящий документ РД 50-680-88 [9], который определяет близкое к ИС понятие автоматизированной системы (АС): *АС – это организационно-техническая система, обеспечивающая выработку решений на основе автоматизации информационных процессов в различных сферах деятельности (управление, проектирование производство и т.д.) или сочетаниях.*

Традиционно выделяемые в информационных системах процессы представлены на рис. 1.

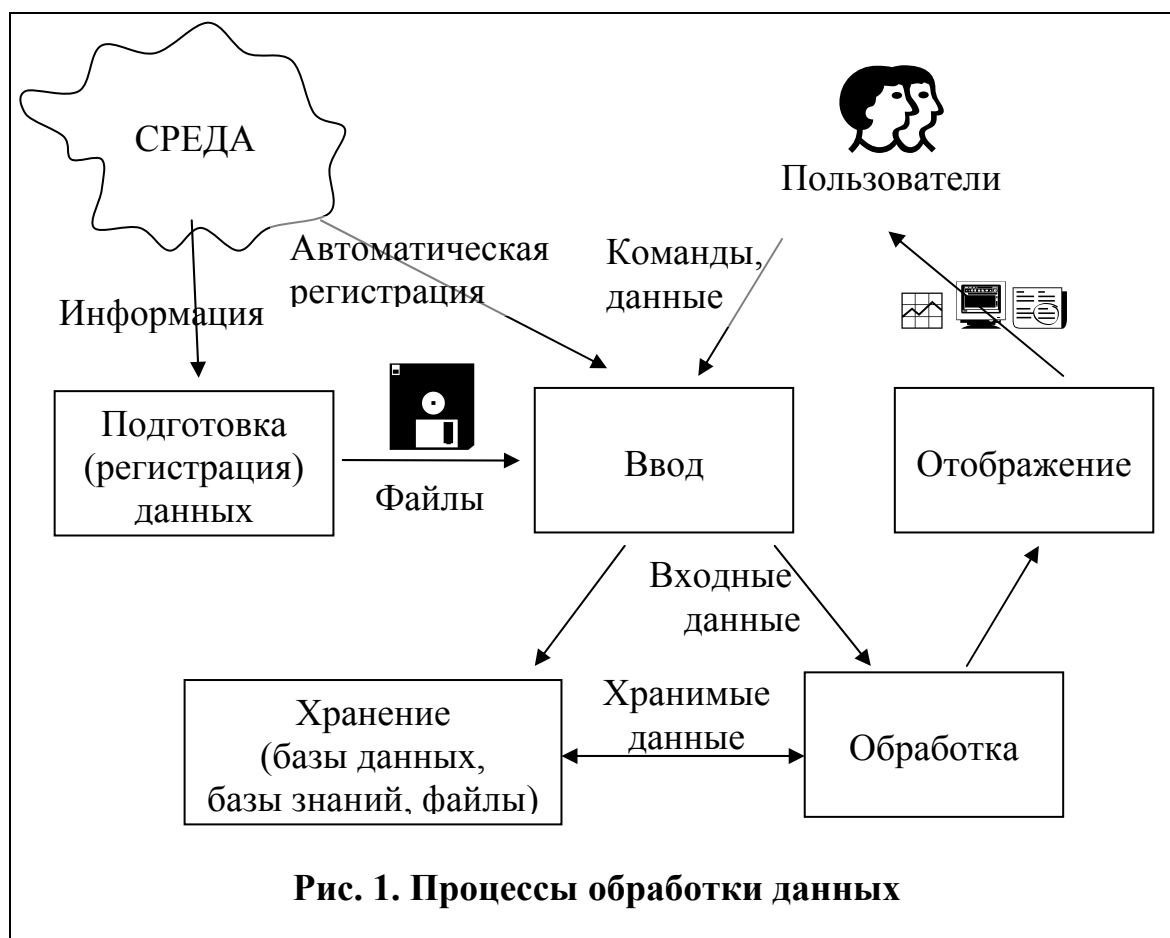


Рис. 1. Процессы обработки данных

Ввод данных пользователями в процессе диалога с компьютером или ручная подготовка данных (перенос данных на машинные носители) являются основными средствами ввода данных в систему и, одновременно, основным источником ошибок в информационной системе. Общей тенденцией является снижение объема ручных операций подготовки и ввода данных за счет применения автоматической регистрации (штрих-коды, магнитные карты, smart-карты, ...).

Накопление и хранение данных всегда было неотъемлемой частью информационных технологий. Современные системы должны обеспечивать одновременный доступ к данным многим пользователям с возможностью разделения прав доступа к данным. Кроме этого, часто требуется распределенное хранение данных на нескольких узлах вычислительной сети. Форма хранения данных должна обеспечивать эффективную обработку данных. Перечисленным выше требованиям удовлетворяют базы данных (БД), доступ к которым обеспечивается соответствующей системой управления базами данных (СУБД).

В современных СУБД выделяют функции оперативного доступа к данным для извлечения и изменения первичных данных (OLTP - OnLine Transaction Processing) и функции аналитической обработки (OLAP – OnLine Analitical Processing).

Другой современной тенденцией является применение баз знаний, которые кроме фактографического материала способны хранить различные правила и накапливать статистические (вероятностные оценки) различных ситуаций. Базы знаний позволяют проверять различные гипотезы и отвечать на широкий круг вопросов, вычислять оценки правдоподобности выводов.

Обработка в современных системах носит распределенный характер. В них выделяют три уровня логики: представления, обработки и доступа к данным. *Логика представления данных* реализуется клиентскими персональными компьютерами и предназначена для ввода пользователем команд и данных и демонстрации ему запрошенных данных и результатов обработки. *Логика обработки данных* обычно сосредоточена на сервере приложений и содержит некоторые общие для всей системы правила и алгоритмы. Часть общезначимой логики обработки данных также может быть передана серверу БД. *Логика доступа к данным* реализуется сервером БД и предназначена для выполнения запросов пользователей в рамках их полномочий.

Отображение данных в наиболее удобной для пользователя форме выделяется в самостоятельную операцию, хотя может рассматриваться как часть обработки данных. Общей тенденцией в построении интерфейса пользователя является увеличение доли графических средств отображения.

В информационной системе можно выделить следующие компоненты:

- пользователи, которые вводят данные и команды, а также поддерживают и развивают ИС;
- инструкции, регламентирующие взаимодействие пользователей между собой и с ИС;
- данные, хранимые и циркулирующие в системе;
- технические средства хранения передачи и обработки данных;
- программы, реализующие алгоритмы обработки, передачи и доступа к данным.

С каждым компонентом системы можно связать несколько видов обеспечения, которые представлены таблице 1.

Таблица 1

Компоненты информационной системы

Компоненты ИС	Компоненты автоматизированной системы в соответствии с ГОСТ 34.003-90
Пользователи	– Пользователь АС – лицо, участвующее в функционировании АС или использующее результаты функционирования АС – Эксплуатационный персонал - персонал, обеспечивающий эксплуатацию вычислительной техники и программ.
Инструкции пользователям	– Организационное обеспечение АС – совокупность документов, устанавливающих организационную структуру, права и обязанности пользователей и эксплуатационного персонала АС. – Правовое обеспечение АС – совокупность правовых норм, регламентирующих правовые отношения, возникающие в процессе эксплуатации АС, и юридический статус результатов функционирования АС. – Эргономическое обеспечение АС – совокупность решений по согласованию психологических, антропометрических и физиологических характеристик и возможностей пользователей АС с характеристиками интерфейса пользователя. – Лингвистическое обеспечение АС – совокупность правил формализации естественного языка, используемых в интерфейсе пользователя АС.
Данные	– Информационное обеспечение – формы существования, объемы и размещение информации в АС.

Компоненты ИС	Компоненты автоматизированной системы в соответствии с ГОСТ 34.003-90
Технические средства хранения передачи и обработки данных	– Техническое обеспечение АС – совокупность всех технических средств, используемых при функционировании АС.
Программы	– Математическое обеспечение – математические методы, модели и алгоритмы, применяемые в АС. – Программное обеспечение АС – программы и программные документы для отладки, проверки работоспособности и функционирования АС.

Существует много подходов к классификации информационных систем. Руководящий документ РД 50-680-88 предлагает следующую классификацию автоматизированных систем в зависимости от сферы автоматизируемой деятельности:

- автоматизированные системы управления (АСУ), которые в свою очередь делят на АСУ предприятием (АСУП) и АСУ технологическими процессами (АСУТП);
- системы автоматизации проектирования;
- автоматизированные системы научных исследований;
- автоматизированные системы обработки и передачи информации;
- автоматизированные системы технологической подготовки производства;
- автоматизированные системы контроля и испытаний.

Классификация требует дополнения видами информационных систем, которые нашли широкое распространение с момента публикации цитируемого документа. Такого рода системами являются, например, информационно-справочные системы, ядром которых являются полнотекстовые базы данных.

В основном, в пособии будут рассматриваться АСУП или экономические информационные системы.

Экономическая ИС - ИС, связанная с управлением некоторым экономическим объектом. В экономическом объекте выделяют (см. рис. 2) систему управления (СУ) и объект управления (ОУ). ОУ выпускает товары или выполняет некоторые услуги. СУ осуществляет воздействие на ОУ для достижения некоторой экономической цели. При этом ОУ получает из внешней среды необходимые энергию и материалы для производства товаров и услуг, а СУ – требования к производству, правила взаимодействия с другими экономическими объектами и сведения о взаимодействии. ОУ

возвращает в среду произведенные товары и услуги, а СУ – сообщения о деятельности экономического объекта.

Экономические ИС характеризуются:

- большим объемом информации,
- цикличностью управления: получения и использования информации,
- многообразием источников и потребителей информации.



Рис. 2. Экономический объект

Экономическая информационная система становится необходимым компонентом контура управления: она хранит все необходимые данные, описывающие состояние и динамику объекта управления, данные управления (планы, графики, расписания, плановые показатели,...), данные, требуемые для учета взаимодействия с окружающей средой.

Относительно автоматизации функций управления определенного уровня (оперативного, тактического, стратегического) можно выделить следующие типы ИС [23] (см. рис. 3):

- СОР – системы обработки транзакций (Transaction Processing/Data Systems – TPS) соответствуют самому низкому уровню управления (иногда его выделяют в отдельный операционный уровень) и связаны с регистрацией транзакций, под которыми здесь понимаются элементарные экономические события (сообщения). Обработка транзакций это даже не уровень управления, а уровень операционистов, выполняющих рутинную работу по оформлению документов (накладных, счетов и т.д.) и регистрации первичных документов. Задача управления на этом уровне заключается в организации делопроизводства, в измерении количества и качества выполняемой работы.
- УИС – управляющие информационные системы (Management Information Systems – MIS) предназначены для автоматизации точно опреде-

ленных, формализованных информационных процедур, связанных, в основном, с уровнем тактического управления. Такие системы используют данные систем обработки транзакций для вычисления показателей деятельности предприятия и выявления отклонений фактических значений от плановых, а также для статистической обработки и прогнозирования.



- АО – системы автоматизации офиса (Office Automation – OA) поддерживают коммуникации, делопроизводство и производительность офиса. Основная проблема здесь состоит в организации распределенной обработки документов в рамках одного отдела (организации), максимально соответствующей структуре и задачам офиса.
- ППР – системы поддержки принятия решений (Decision Support Systems – DSS) применяют для содействия принятию решений плохо формализованных задач в условиях недостатка информации и большой неопре-

деленности. Здесь могут применяться методы извлечения информации по запросам пользователя, позволяющие ему оценивать ситуацию с различных точек зрения, сценарии вида "что, если", методы имитационного моделирования. Системы этого типа связаны с высшими уровнями управления, достаточно специфичны и в технологическом плане требуют применения компонентов искусственного интеллекта.

- ЭС – экспертные системы (Expert Systems – ES) предназначены для выделения плохо формализуемых знаний и навыков специалистов – экспертов и для последующего тиражирования и использования экспертных систем в качестве консультантов для анализа и принятия решений. Экспертные системы находят широкое применение для диагностики, анализа различных ситуаций и оценки решений. Системы такого типа также применяются, в основном, на высших уровнях управления.
- ВР – информационные системы для высшего руководства (Executive Information Systems – EIS) предназначены для удовлетворения информационных запросов высшего руководства. Они предоставляют информацию в виде набора наиболее значимых для управления сведений, показателей и отклонений плановых и фактических значений. Однако, при необходимости они способны показать и детальную информацию, объясняющую значение того или иного показателя. В основном они предназначены не для ввода новых данных, а для извлечения информации по различным аспектам деятельности предприятия.

Другой основой классификации ИС является функциональное назначение, которое обычно совпадает с основными функциями предприятия и делением на подразделения (см. рис. 4). Очень часто эта классификация лежит в основе разделения на взаимодействующие подсистемы.

Отдел маркетинга планирует спектр товаров и услуг и способы продвижения их на рынке. В том числе определяются потребительские свойства товаров и прогнозируется объем продаж.

Отдел сбыта непосредственно продвигает товар на рынок, организует рекламные компании, находит покупателей, реализует технологию продаж и предоставления услуг.

Управление производством обычно включает конструкторскую и технологическую подготовки производства, календарное планирование, управление процессами производства и учет готовой продукции. Уже этот перечень задач показывает, какой большой объем данных необходимо хранить и обрабатывать для автоматизации этих функций.

Материально-техническое снабжение включает планирование и организацию закупок, необходимых для производства материалов и комплектующих, хранение и передачу их в производство.

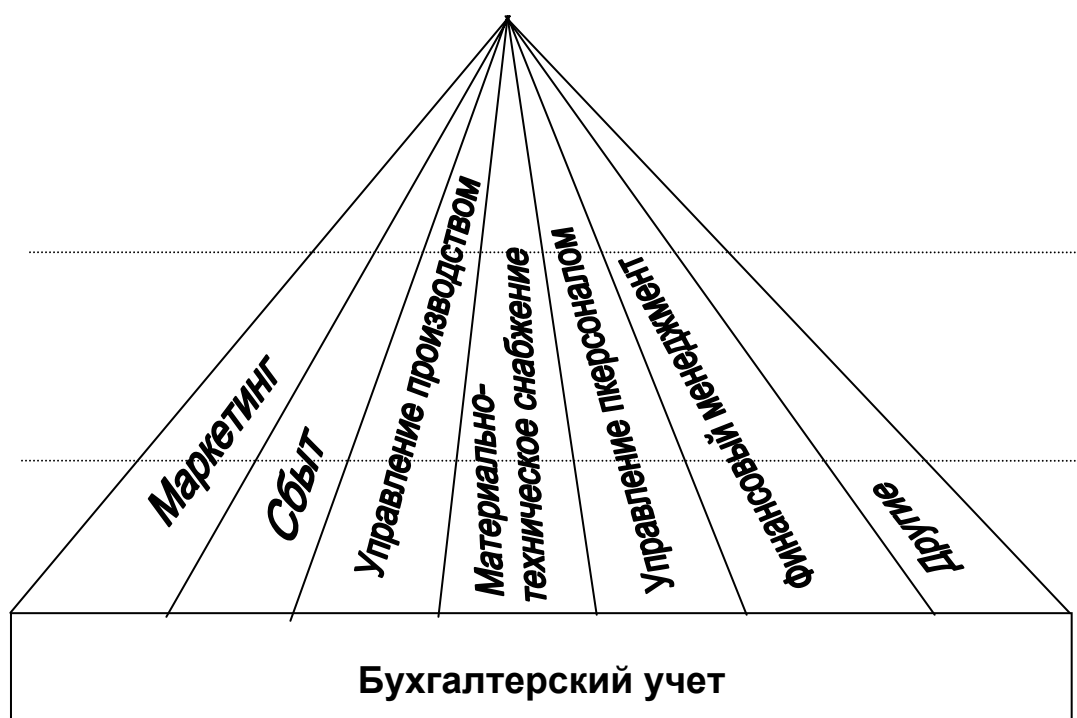


Рис. 4. Классификация ИС по функциональному признаку (подсистемы)

Управление персоналом в современных системах становится все более значимым. Это связано с осознанием того, что человеческий ресурс является наиболее важным для успешной деятельности предприятия. Управление персоналом теперь должно предусматривать не только учет состояния и движения кадров, но и планирование карьеры, обучение и повышение квалификации персонала, оптимальный подбор кадров и многое другое.

Финансовый менеджмент занимается оптимизацией денежных потоков, измерением и оценкой финансовой деятельности и состояния предприятия.

Подсистема бухгалтерского учета выделена на рисунке как основание, так как бухгалтерия занимается сбором данных по всем сторонам деятельности предприятия, измерением объемов производства, продаж, затрат всех ресурсов в стоимостном и натуральном измерениях.

2. Жизненный цикл информационной системы

Жизненный цикл информационной системы (программы) – это процесс ее построения и развития. В жизненном цикле ИС выделяют стадии, а каждую стадию разбивают на этапы. Для каждого этапа описывают выполняемые работы и участие в них системных аналитиков, программистов и пользователей.

Существует несколько стандартов на жизненный цикл ИС. Один из них – это стандарт ГОСТ 34.601-90 [9], который предусматривает следующие стадии и этапы создания автоматизированной системы (АС):

1. Формирование требований к АС.
 - 1.1. Обследование объекта и обоснование необходимости создания АС.
 - 1.2. Формирование требований пользователя к АС.
 - 1.3. Оформление отчета о выполнении работ и заявки на разработку АС.
2. Разработка концепции АС.
 - 2.1. Изучение объекта.
 - 2.2. Проведение необходимых научно-исследовательских работ.
 - 2.3. Разработка вариантов концепции АС и выбор варианта концепции АС, удовлетворяющего требованиям пользователей.
 - 2.4. Оформление отчета о проделанной работе.
3. Техническое задание.
 - 3.1. Разработка и утверждение технического задания на создание АС.
4. Эскизный проект.
 - 4.1. Разработка предварительных проектных решений по системе и ее частям.
 - 4.2. Разработка документации на АС и ее части.
5. Технический проект.
 - 5.1. Разработка проектных решений по системе и ее частям.
 - 5.2. Разработка документации на АС и ее части.
 - 5.3. Разработка и оформление документации на поставку изделий для комплектования АС и (или) технических требований (технических заданий) на их разработку.
 - 5.4. Разработка заданий на проектирование в смежных частях проекта объекта автоматизации.
6. Рабочая документация.
 - 6.1. Разработка рабочей документации на систему и ее части.
 - 6.2. Разработка и адаптация программ.
7. Ввод в действие.
 - 7.1. Подготовка объекта автоматизации к вводу АС в действие.
 - 7.2. Подготовка персонала.
 - 7.3. Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями).
 - 7.4. Строительно-монтажные работы.
 - 7.5. Пуско-наладочные работы.
 - 7.6. Проведение предварительных испытаний.
 - 7.7. Проведение опытной эксплуатации.
 - 7.8. Проведение приемочных испытаний.

8. Сопровождение АС.

8.1. Выполнение работ в соответствии с гарантийными обязательствами.

8.2. Послегарантийное обслуживание.

Эскизный, технический проекты и рабочая документация – это, по существу, последовательное построение все более точных проектных решений по всем вида обеспечения информационной системы. Допускается исключать стадию "Эскизный проект" и отдельные этапы работ на всех стадиях, объединять стадии "Технический проект" и "Рабочая документация" в "Технорабочий проект", выполнять отдельные этапы работ до окончания предшествующих стадий, разрешается параллельное по времени выполнение этапов работ и включение дополнительных этапов работ.

Широко применяемый зарубежный стандарт [11] предусматривает следующие стадии обобщенной модели жизненного программных средств:

1. Системный анализ и разработка спецификации. Спецификация содержит требования к программе и по содержанию аналогична техническому заданию.
2. Предварительное (внешнее) проектирование. На этой стадии принимаются базовые решения по архитектуре программного средства: выделение самостоятельных программных компонент, определение регламента их взаимодействия, функций, входных, выходных и хранимых данных, интерфейса пользователя.
3. Детальное (внутреннее) проектирование – детализация структур входных, выходных и хранимых данных, построение алгоритмов обработки данных, структуры и внутреннего интерфейса программных компонент.
4. Кодирование и отладка компонентов – создание программной реализации компонентов программного средства, выявление и исправление ошибок.
5. Интеграция и комплексная отладка – объединение компонентов в единое программное средство, выявление и исправление ошибок взаимодействия компонент.
6. Испытание и документирование – проверка работоспособности и создание различных видов документации для развития, сопровождения и использования программного средства.
7. Поддержка эксплуатации – выполнение штатных процедур (резервное копирование, проверка работоспособности, контроль целостности данных,...)
8. Сопровождение – выявление и устранение ошибок, развитие и адаптация программ, измерение и увеличение производительности.

Жесткая последовательность стадий и этапов разработки ИС известна как каскадная схема и была сформулирована в 60-е годы. Достоинствами ее являются:

- управляемое построение ИС,
- юридически зафиксированная ответственность участников разработки: как разработчиков, так и заказчиков,
- возможность применения структурных методов разработки,
- развитие ИС в соответствии с определенными функциями и бюджетом.

В то же время каскадной методологии разработки присущи следующие недостатки:

- избыток согласующей документации,
- в начале разработки предполагается идеальный прогноз будущей ситуации,
- последовательный характер выполнения работ,
- обнаружение ошибок откладывается на последний этап,
- неспособность произвести работающую ИС за короткое время.

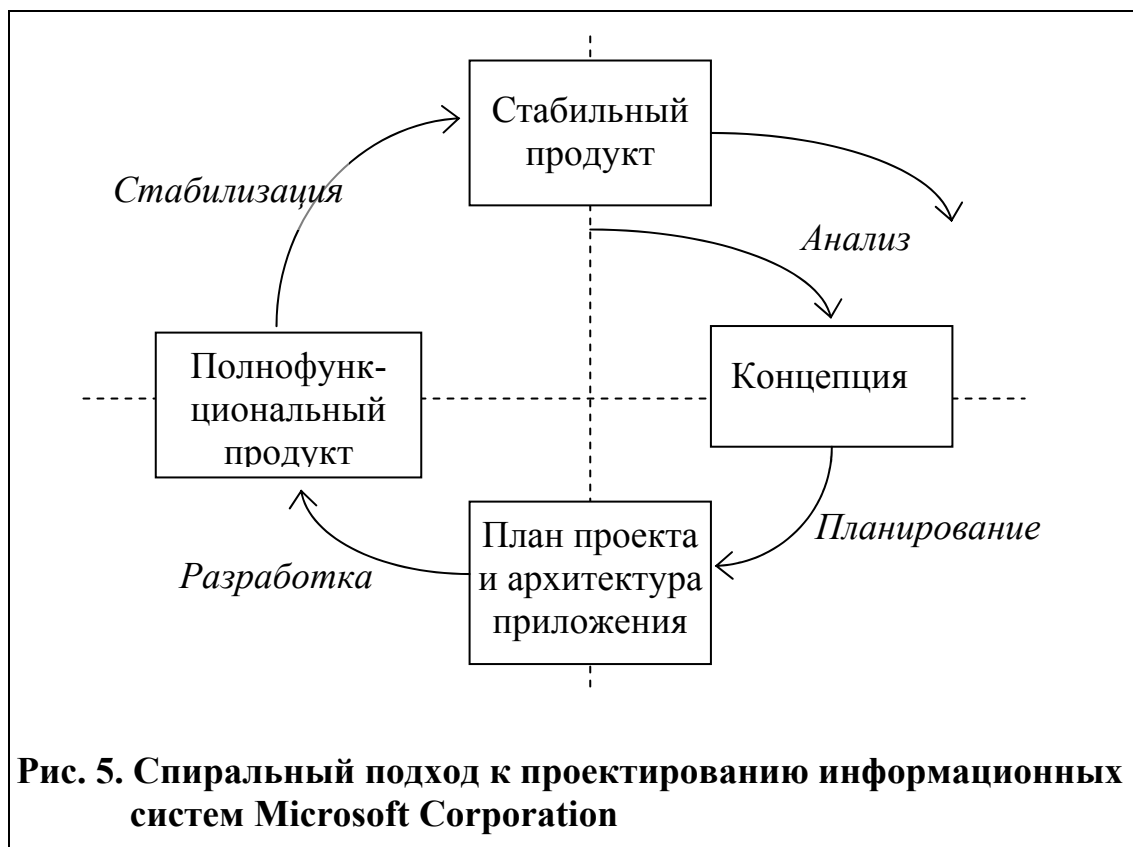
Излишняя жесткость каскадного подхода мешала оперативно вносить изменения в проект при обнаружении ошибок или изменений условий функционирования на последних этапах проектирования. Для устранения недостатков каскадного проектирования в 70-х годах был предложен итерационный подход, который, не меняя содержания работ, предлагает по результатам каждого этапа переосмысление и изменение всего проекта. Гибкость процесса проектирования была достигнута, но ценой постоянного пересмотра проектных решений, планов и бюджетов.

В 80-х годах появился спиральный подход, который заключается в построении работающего ядра или прототипа системы с последующими доработками, модификациями, а главное – развитием и расширением функциональных возможностей.

Этот подход (см. рис. 5) по стандартам фирмы Microsoft включает следующие фазы [1]:

- Анализ – разработка концепции продукта.
- Планирование – создание подробного плана проекта и архитектуры приложения.
- Разработка – выработка полнофункционального продукта.
- Стабилизация – доработка продукта до стабильного состояния, готового к развертыванию.

Продукт проходит все четыре фазы неоднократно с целью последовательного уточнения характеристик и архитектуры для изменения и увеличения количества автоматизированных функций.



Спиральный подход во многом снимает проблемы каскадного проектирования. В приемлемое время заказчик получает первую очередь продукта, может оценить полученный результат и принять обоснованное решение об изменении и дальнейшем развитии продукта.

В то же время спиральный подход порождает ряд проблем:

- задержки в получении полнофункциональной информационной системы,
- возможность получить «никогда не заканчивающуюся» разработку,
- сложность в определении стоимости всех итераций проекта,
- постоянные обновления приводят к отсутствию стабильности продукта.

Описанная методология проектирования является процедурно- или функционально-ориентированной. Сначала выделяются процедуры или функции, подлежащие автоматизации, затем для них определяются входные и выходные данные. Последовательное применение этого подхода зачастую приводит к набору несвязанных между собой подсистем, а это, в свою очередь, – к дублированию и несогласованности данных разных подсистем. Альтернативным ему является информационно-ориентированный подход [22]. В этом подходе основой проектирования являются данные, необходимые для обеспечения автоматизируемых функций. При этом рассматриваются все информационные потребности предприятия, даже те, автоматизация которых планируется в последнюю очередь.

Если процедурно-ориентированный подход предназначен для выполнения отдельных проектов, то для создания корпоративных ИС более подходящим является информационно-ориентированный подход к проектированию. Этот подход рассматривает проектирование структур данных как первоочередную задачу, соблюдая, тем не менее, баланс "данные-процессы". Информационно-ориентированный подход использует те же методы, которые были разработаны для процедурно-ориентированного подхода. Более того, стадии и этапы разработки приложения остаются сходными, за исключением первых этапов проектирования. Информационно-ориентированный подход предполагает следующую последовательность разработки:

1. Стратегическое планирование ИС.
2. Выделение и анализ управленческих подсистем.
3. Выделение наиболее приоритетных подсистем для разработки.
4. Разработка подсистем.

Наиболее значимым отличием данного подхода является построение описания всех данных организации на основании самых общих сведений об их использовании, даже если ближайшей целью является разработка только одного приложения. Таким образом, сначала исследуются данные такой ИС, которая охватывает все сколько-нибудь значимые виды деятельности организации, и затем выделяются структуры данных для подсистем, выбранных для автоматизации в первую очередь. Последний этап "Разработка подсистем" выполняется с применением методов и приемов процедурно-ориентированного подхода. Достоинствами информационно-ориентированного подхода являются соответствие проектируемой ИС стратегии бизнеса и высокая степень ориентации на пользователя.

В [23] описан подход, который объединяет информационно-ориентированный подход с традиционным и отличается от последнего первым этапом (см. рис. 6). Задачей этого этапа является создание представления об информационной системе с учетом потребностей всех подразделений и на этой основе общей информационной модели.

Планирование. Планирование ИС выполняется для всей организации или ее крупного самостоятельного подразделения. Целью планирования является максимизация соответствия системы задачам бизнеса и оптимизация технологии их решения. Для этого, критически переосмысливаются существующая система управления и информационная система для выделения приоритетных направлений автоматизации. Результатом планирования являются архитектура ИС (архитектура данных, функций, организационная архитектура, архитектура технических средств), список приоритетных приложений, план развития ИС. Планирование ИС как часть планирования бизнеса выполняется непрерывно – обычно раз в год. Толчком к вы-

полнению планирования также может послужить появление новых информационных технологий, или реструктуризация предприятия.



Анализ. Анализ выполняется в рамках некоторого выделенного проекта – для приложения или системы связанных приложений. На основании оценки и описания существующей ИС, потребностей управления, новых возможностей для бизнеса, изменений в правилах ведения дел вырабатывается набор требований и приоритетов для приложений со стороны управления. Результатом является утвержденный список требований (техническое задание на разработку ИС).

Конструирование. На основании требований к системе с учетом мнений и рекомендаций пользователей вырабатываются принципиальные информационные решения в части структур хранения данных, методов и алгоритмов обработки, структуры сети, интерфейса пользователей. В результате этих работ создается документация разработчика, на основе которой выполняется программирование.

Реализация. На этой стадии создаются вычислительная сеть, структуры данных, программы и документация к ним, инструкции по использова-

нию программ, выполняется обучение пользователей и готовые программы запускаются в эксплуатацию.

Сопровождение. Сопровождение заключается в поддержании в рабочем состоянии программного обеспечения. На этой стадии выполняются консультации и дополнительное обучение пользователей, восстановление после сбоев, исправление ошибок и адаптация программы к изменениям в условиях эксплуатации.

На каждой стадии выполняются следующие работы:

- сбор данных, для этого выполняются исследования и интервью, проводятся собрания и семинары;
- каждый этап работ документируется и по окончании работ проводится презентация для подведения итогов и принятия решений;
- выполняются оценки временных, трудовых, стоимостных и иных затрат на разработку и эксплуатацию ИС, а также выгод от ее применения;
- анализируется целесообразность ИС, так что на любом этапе нецелесообразная разработка может быть прекращена;
- управление проектами (расписанием, бюджетом, качеством работ и их исполнением).

Между стадиями различных моделей жизненного цикла можно провести аналогию (см. табл. 2). Разница заключается в том, какие вопросы проектирования считаются ключевыми. При этом стратегия спирального проектирования в совокупности с информационно-ориентированным подходом в большей степени отвечает современному уровню методологии проектирования. В дальнейшем изложение будет ориентировано на стадии последнего подхода, хотя методы проектирования могут применяться достаточно независимо от выбранной модели жизненного цикла.

Таблица 2

Жизненный цикл информационной системы

Стадии жизненного цикла ИС	Стадии создания АС в соответствии с ГОСТ 34.601-90	Документы
Планирование	1. Формирование требований к АС.	Отчет и заявка на создание АС
	2. Разработка концепции АС.	Отчет о НИР
Анализ	3. Техническое задание.	Техническое задание

Стадии жизненного цикла ИС	Стадии создания АС в соответствии с ГОСТ 34.601-90	Документы
Конструирование	4. Эскизный проект.	Схема организационной структуры, схема функциональной структуры, схема комплекса технических средств, модель данных "сущность-связь"
	5. Технический проект	Описание функций и задач, описание информационного обеспечения, описание математического обеспечения
Реализация	6. Рабочая документация.	Общее описание системы, описание технологического процесса обработки данных, описание программного обеспечения, руководство пользователя
	7. Ввод в действие.	План-график работ, протокол испытаний, протокол рассогласования, акты приемки в опытную и промышленную эксплуатацию
Сопровождение	8. Сопровождение АС.	

Различные этапы разработки ИС требуют различного количества ресурсов. На рис. 7 представлен примерный график изменения трудозатрат на разных этапах развития ИС [20]. График составлен с учетом изменений требований к ИС в процессе проектирования, что приводит к пересмотру результатов всех предыдущих этапов разработки ИС.

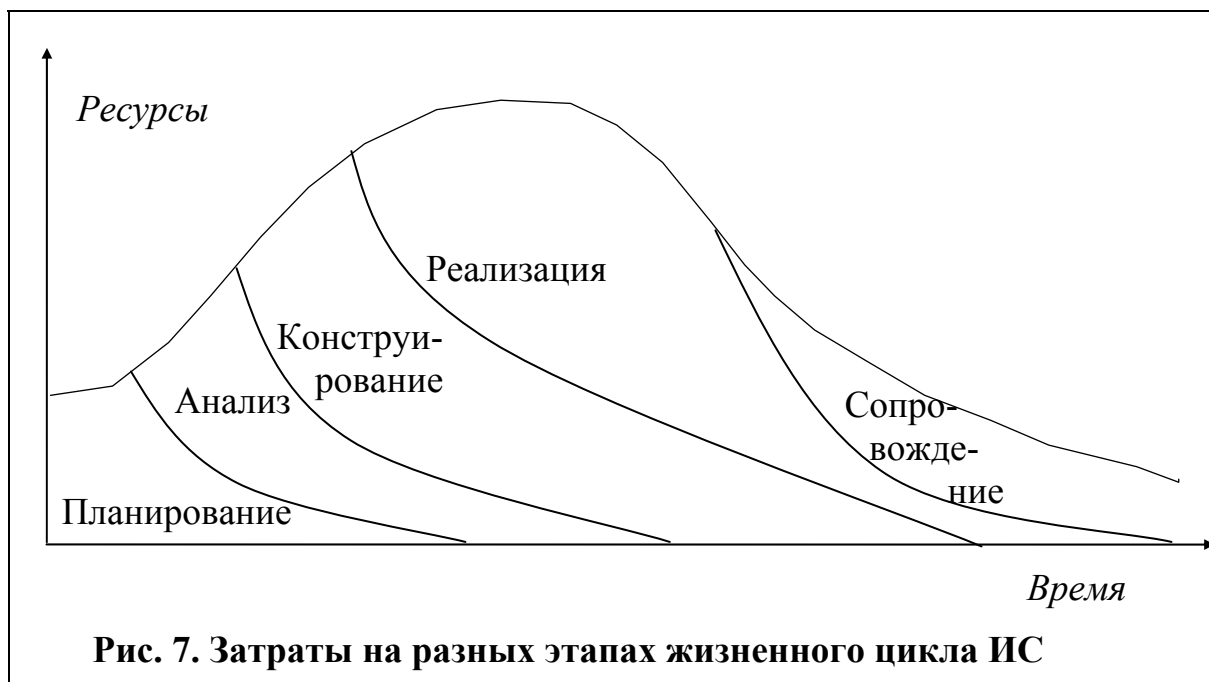
Объединению стадий (или «размыванию» границ между стадиями) способствуют современные технологии проектирования:

1. Joint Application Development (JAD) – организация совместной работы пользователей, менеджеров и разработчиков ИС для определения требований, приоритетов и, собственно, проектирования ИС.

2. Prototyping – построение и/или использование прототипов для определения требований, приоритетов и проектирования.

3. Rapid Application Development (RAD)- методы быстрой разработки программ.

4. Computer Aid System Engineering (CASE) – автоматизированные технологии разработки ИС.



Знание некоторых принципов легко возмещает незнание некоторых фактов.

К. Гельвеций

3. Принципы разработки информационных систем

Принципы разработки информационных систем, с одной стороны, являются результатом осмысления большого опыта проектирования, с другой — представляют неформальное дополнение стандартов. Следование принципам требует внутренней дисциплины всей команды и каждого разработчика в отдельности, но все затраты окупаются меньшим количеством переделок и качеством выполнения проектирования. Нарушение принципов приводит к проблемам создания и эксплуатации системы.

Периодически появляются новые списки принципов, принадлежащие различным коллективам и отражающие изменения в технологии проектирования. Исторически первыми в СССР появились принципы проектирования АСУ, предложенные академиком В.М.Глушковым [2, 7]:

- 1) *Принцип системности* заключается в повсеместном применении системного подхода:
 - установление основных направлений хозяйственной деятельности системы и ее основных функций;
 - выявление структурных элементов и взаимосвязей между ними:
 - макроанализ: выделяются функции и связи элемента,
 - микроанализ: выделяется и изучается структура элемента;

- формирование многоуровневой иерархии: система, подсистемы,
- 2) *Принцип непрерывного развития* предполагает постоянное изменение и наращивание возможностей ИС. Несомненно, что высокая адаптивность ИС остается актуальной и сейчас.
 - 3) *Принцип совместимости* заключается в реализации информационного обмена между системами и подсистемами. В настоящее время он нашел свое воплощение в построении корпоративных ИС на основе совместного использования распределенных данных.
 - 4) *Принцип стандартизации и унификации* предусматривает широкое использование стандартных решений и компонентов.
 - 5) *Принцип эффективности* - создание АСУ должно быть экономически выгодным.

Кроме перечисленных общих принципов были предложены более детальные частные принципы:

- 1) *Принцип декомпозиции* - разделение системы на части для последующего анализа и реализации.
- 2) *Принцип первого руководителя* говорит о том, что в организационном плане возглавлять разработку должен первый руководитель. В современной трактовке принцип значительно шире - требуется привлечение и интерес к разработке ИС всех ключевых пользователей, а не принуждение к сотрудничеству с разработчиками со стороны руководителя.
- 3) *Принцип новых задач* заключается в отказе от механической передачи функций обработки информации от человека компьютеру. Вместо этого должно быть найдено такое применение вычислительной техники, которое позволило бы решать качественно новые задачи (например, оптимизации загрузки оборудования).
- 4) *Принцип автоматизации информационных потоков и документооборота* предусматривает использование технических средств на всех этапах создания и обработки документов.
- 5) *Принцип автоматизации проектирования* предполагал использование информационных технологий для создания, ведения и доступа к проектной документации, автоматическое генерирование заготовок проектных документов, структур данных, программ по проектным описаниям. Сейчас эта идея реализована в CASE - средствах.

Наконец, в общий свод вошли организационно- технологические принципы:

- 1) *Принцип абстрагирования* заключается в выделении существенных для решения задачи аспектов системы и исключении несущественных.
- 2) *Принцип формализации* заключается в получении строго формальных описаний. Для этого были разработаны и успешно применяются не-

сколько схем, систем записи, языков для отображения различных проектных решений.

- 3) *Принцип концептуальной общности* заключается в следовании единой методологии построения системы.
- 4) *Принцип непротиворечивости и полноты* заключается в построении всех необходимых для решения задач компонентов и в отсутствии противоречий между ними. Принцип должен соблюдаться при разработке всех моделей и описаний на каждом этапе проектирования.
- 5) *Принцип независимости данных* - данные должны проектироваться независимо от процессов их обработки в соответствии с информационными потребностями системы. Последовательное применение этого принципа привело к объединению всех экземпляров хранимых данных в единую базу данных и созданию СУБД как единого средства доступа к данным.
- 6) *Принцип структурирования данных* предполагает выделение единиц данных и связей между ними, описание хранимых структур и запросов обработки данных на единой методологической основе.
- 7) *Принцип доступа конечного пользователя* заключается в наличии средств оперативного доступа конечного пользователя к данным системы.
- 8) *Принцип однократного ввода* был очень важен для первых ИС, в которых в условиях отсутствия совместного использования данных одна и та же информация вводилась неоднократно в разных подсистемах, что приводило к созданию и накоплению противоречий. Ввод данных и поныне остается одной из самых сложных задач и источником различных ошибок, хотя в этом отношении накоплен большой опыт использования технических средств (магнитные карточки, штрих-коды) и организационных мероприятий (специальные методы проверки, совмещение ввода данных и формирования первичного документа).

Многие из принципов Глушкова не потеряли актуальности, несмотря на то, что были сформулированы в 70-х годах, другие (как принцип доступа конечного пользователя) стали очевидной реальностью. Предлагаемый далее свод признаков отражает взгляд зарубежных специалистов на проектирование информационных систем [23].

1. *Привлекайте пользователей.* Это позволяет обнаруживать ошибки проектирования на самых ранних этапах, уменьшать затраты на обучение и адаптацию системы. Важно, чтобы конечные пользователи с самого начала считали ИС своим детищем. Существуют специальные методы привлечения пользователей, например, Joint Application Development.
2. *Используйте ИС для решения проблем.* Проблемный подход предусматривает следующие этапы решения:

- 1) Идентификация проблем, возможностей или изменения условий функционирования системы (например, изменение законодательства или правил ведения бухгалтерского учета).
 - 2) Выявление причин, вызвавших проблему, и последствий решения проблемы.
 - 3) Разработка требований к решению.
 - 4) Выявление альтернативных решений.
 - 5) Выбор наилучшего решения.
 - 6) Реализация решения.
 - 7) Оценка результатов решения.
3. *Устанавливайте стадии и этапы разработки ИС и описывайте работы на каждом этапе.* Следует выбрать наиболее подходящую модель жизненного цикла и сформулировать требования к составу и качеству работ.
 4. *Устанавливайте стандарты для разработки и документирования.* Стандарт должен описывать работу, ответственность исполнителей, управление разработкой, процедуры проверки качества. Отсутствие или несоблюдение стандартов приводит к ситуациям, когда в случае ухода одного из разработчиков теряется часть проектных решений.
 5. *Рассматривайте ИС как капитальное вложение.* Это означает, что альтернативные решения должны оцениваться с помощью различных методов экономического анализа: затраты - прибыль, цена - качество, совокупная стоимость владения и других [19]. Однако, если затратная часть проектирования рассчитывается достаточно просто, то прибыль от внедрения информационных технологий оценить сложно, так как она может быть связана с изменением качества управленческих решений. Тем не менее, сравнение с экономической точки зрения различных вариантов решения возможно и необходимо.
 6. *Не бойтесь прекратить или переосмыслить разработку.* Необходимо предусмотреть проверки целесообразности проекта на каждой стадии проектирования. В каждый такой момент рассматриваются вопросы переоценки стоимости проекта, его соответствия потребностям бизнеса и при обнаружении нецелесообразности проекта принимается болезненное, но совершенно необходимое решение о прекращении работ.
 7. *Применяйте принцип «разделяй и властвуй» для управления сложностью и последовательностью выполнения работ.* Большая сложность является отличительной чертой современных экономических ИС, охватывающих большую часть процессов управления предприятием. Поэтому очень важно строить информационную систему по частям, тщательно планируя разбиение на контуры управления (подсистемы) и регламент их взаимодействия.

8. *Создавайте системы, способные расти и изменяться.* Ясно, что "жесткие" системы придется заменять всякий раз при изменении условий или технологии управления. С другой стороны большие адаптационные возможности, во-первых, сложны в настройке и требуют специальных знаний, во-вторых, увеличивают стоимость разработки. Поэтому особую значимость приобретают специальные методы проектирования, позволяющие отделять данные от программ доступа и обработки, строить систему из независимых модулей, соблюдать соответствие программ и их описаний.

При всей разнице формулировок очевидно, что различные наборы принципов содержат по-разному сформулированные похожие рекомендации. Отличия обусловлены пристрастиями авторов и технологическими изменениями в области информационных технологий. Очевидна и некоторая противоречивость принципов. Так, адаптивность требует от программного обеспечения большей гибкости и функциональности, чем необходимо в момент разработки и это, несомненно, увеличивает стоимость работ. В целом принципы являются концентрированным отражением культуры разработки информационных систем.

Мудреца спросили: "Скажи, можешь ли ты съесть целого слона?" "Могу, - ответил мудрец, - если буду его есть по кусочкам".

4. Методология разработки

На разных этапах разработки ИС приходится решать общие в методическом отношении задачи:

- Управление сложностью разработки - невозможно сразу создать (и описать) все детали информационной системы – приходится выделять и разрабатывать более или менее самостоятельные блоки.
- Задача определения границ является своеобразным продолжением структурных методов – при выполнении любой части проекта требуется тщательно определять область разработки (внешние границы) и уровень детализации (внутренние границы).
- Задача выработки и соблюдения требований к качеству и готового изделия, и процесса его разработки.

Известны и широко применяются следующие подходы к управлению сложностью:

1. Декомпозиция «сверху-вниз». Исходное целое разбивается на некоторые части, каждое из которых в дальнейшем подвергается аналогичному делению до получения некоторых простейших единиц.

2. Синтез «снизу-вверх» противоположен предыдущему подходу – из простых частей пытаются скомбинировать более сложные блоки, и так далее до получения требуемого результата.
3. Подход «от центра к границам» (Centre-out) комбинирует два первых подхода. Изучение начинается с некоторого среднего по уровню детализации объекта. Исследуется его структура (декомпозиция) и связь с другими объектами (синтез). Подход применяется при изучении различных структур управления.

Неправомерно противопоставление этих подходов - на самом деле они имеют различную область применения. Пошаговая декомпозиция применяется когда объект и его свойства достаточно точно определены. Подход «снизу-вверх» подходит для создания инструментальных средств, библиотек подпрограмм, предназначенных для решения широкого (и не вполне определенного) круга задач в некоторой предметной области.

Пошаговая декомпозиция является наиболее распространенной методологией решения задач и породила множество методов, которые называют структурными:

1. Структурное программирование – ограничение управляющих конструкций следованием, выбором и циклом (отказ от Go To) с возможностью их вложения друг в друга. Эти ограничения в сочетании с определением процедур и функций является стройной концепцией пошаговой декомпозиции в программировании.
2. Структурный анализ или процедурно-ориентированный подход. Выполнение задачи представляют в виде иерархической системы функций. На верхнем уровне иерархии находится функция, решающая задачу, которая делится на более элементарные. Разбиение прекращают при получении "неделимых" в некотором смысле функций. Для функций описываются входные и выходные данные, которые также могут быть структурированы.
3. Моделирование данных – описание системы данных в виде совокупности единиц информации. При этом выделяют элементарные единицы, такие как числа, строки, логические данные и другие, и составные единицы, такие как массивы (наборы пронумерованных единиц одного типа) и структуры (наборы единиц разных типов).
4. Объектные модели. Объекты хранят данные и предоставляют методы для доступа и обработки хранимых данных. В отношении объектов можно рассматривать несколько иерархий (декомпозиций). Одну из них составляет иерархия классов, реализованная через механизм наследования – определения нового класса при помощи модификации существующего класса. Другая иерархия описывает вложенность объектов

друг в друга. Третья определяет взаимодействие объектов в процессе выполнения программы.

Приведенный список отражает только некоторые распространенные структурные методы.

Определение границ, т.е. содержания и уровня детализации является важным с методологической точки зрения – слишком часто встречаются описания, в которых либо не указаны значимые для понимания моменты, либо перегруженные несущественными деталями. При определении границ описания выделяют:

- внешние границы, определяющие описываемую часть реального мира,
- внутренние границы, задающие уровень детализации.

Границы описания рекомендуется устанавливать до начала исследования и приводить во введении к описанию. Для сложных объектов, каковыми являются ИС, должно существовать несколько описаний с разными границами. Например, внешней границей описания ИС фирмы будут все виды деятельности и информационные источники, а внутренней – подсистемы и информационные блоки.

Требования к качеству должны быть составной частью процесса проектирования. Плохим вариантом является сначала выполнение работы, а затем проверка ее качества. В этом случае объем работ на переделку может существенно возрасти. Причины этого заключаются в запаздывании времени обнаружения ошибок и несоответствия требуемому уровню качества (см. рис. 8). Так, наиболее значимые ошибки в определении требований обнаруживаются на самых последних этапах разработки, когда с продуктом начинают работать конечные пользователи.

Отдельно рассматривают качество конструкции и качество изготовления (см. табл.3).

Для каждого вида работ важны следующие характеристики качества:

1. Соответствие ожиданиям/требованиям пользователя. Качество соответствия обеспечивается:
 - участием пользователя в планировании и конструировании ИС,
 - применением технологий и систем записи, соответствующих подготовке пользователя,
 - использованием специальных приемов и технологий общения.
2. Гибкость ИС. Гибкости добиваются посредством независимости систем и частей ИС.
3. Универсальность (portable). Способами достижения универсальности являются разработка более сложных процедур чем требуется в данном проекте с учетом прогноза изменения ИС и тщательное документирование разработки.



В отношении моделей, используемых в процессе проектирования должны соблюдаться следующие нормы:

1. Модели должны соответствовать будущему реальному миру. Для этого необходимо обеспечить следующее:
 - понятность моделей для пользователей,
 - реальное участие пользователей в разработке: все аспекты ИС, которые касаются пользователей обсуждаются и согласовываются с ними,
 - построение моделей на сопоставительном анализе "как есть" и "как должно быть".
2. Каждая модель должна быть полной и непротиворечивой.
3. Набор моделей должен быть полным и модели не должны противоречить друг другу. Для этого модели должны формально следовать одна из другой, а подсистемы моделей должны соответствовать друг другу.

Таблица 3

Факторы, влияющие на качество информационной системы

	Соответствие ожиданиям/ требованиям поль- зователя	Гибкость – расширяемость – возможности сопро- вождения	Универсальность
Качество конструкции	– Степень соответствия ожиданиям/требованиям пользователя – Изменение требований в процессе разработки	– Независимость частей (высокая внутренняя связанность компонент каждой части, низкая степень зависимости частей друг от друга) – Нормализация структур данных – Структурированность процессов обработки данных	– Применение стандартных технологий и систем записи – Понятность документации и конструкции в целом – Управление изменениями
Качество изготовления	– Производительность – Доступность описания проектных решений – Количество ошибок – Количество ненормальных завершений	– Модульность – Структурный код	– Применение стандартных технологий и систем записи – Читаемость кода – Управление изменениями

Вы уже дошли до состояния, когда у вас нет времени, чтобы разрешить те проблемы, которые отнимают у вас все время???

Марк Дэвидсон

5. Планирование развития информационной системы

Целью планирования ИС является построение информационной системы организации, которая обеспечивала бы наиболее эффективное решение задач управления. Основной методикой является сопоставление текущей

ситуации «как есть» и ситуации «как должно быть» и выделение в результате такого анализа приоритетных проектов разработки информационных технологий и приложений, наиболее полезных для бизнеса.

Распространенная ошибка проектирования заключается в построении ИС, ориентированной на текущую ситуацию. При этом ИС фиксирует все недостатки существующей системы управления, делая ее изменение еще более трудным. Именно по этой причине появилось следующее мнение о пользе автоматизации: "вред или бесполезность".

В процессе планирования ИС важно учитывать следующее:

- Понимание ситуации в целом важнее детального описания.
- Целью автоматизации является в большей степени инновация управленческой деятельности и в меньшей - автоматизация существующих технологий (что как правило приводит к плохим результатам, полученным в результате больших затрат).
- Требуется больше усилий для подготовки конечных пользователей к инновациям.
- Детальное изучение может быть выполнено при необходимости.

На стадии планирования ИС выделяют два этапа:

1. Изучение и анализ организации.
2. Планирование архитектуры ИС и выделение наиболее актуальных проектов.

5.1. Изучение и анализ организации

Задачей первого этапа является изучение методов и технологии управления в целом, и возможностей улучшения управления на основе компьютерных технологий. Эта задача является очень сложной, неформализуемой и требует высокой квалификации в области анализа и конструирования бизнес-процессов, а также хорошей эрудиции в области информационных технологий.

Начинается изучение с общего описания организации, в которое рекомендуется включить следующие пункты:

1. Миссия (место организации в общественном разделении труда), цели (набор значений показателей, таких как доля или сегмент рынка, которые организация хотела бы достигнуть), задачи (мероприятия, которые должны привести к достижению целей).
2. Методы достижения целей, решения задач (если целью является захват доли рынка, то какими средствами фирма стремится это сделать).
3. Характеристика организации:
 - виды деятельности и их взаимосвязи и значимости;
 - применяемое оборудование;

- внешняя среда (поставщики, потребители, органы государственного регулирования) и взаимодействие с ней;
 - политика организации;
 - технология планирования деятельности;
 - прогноз развития.
4. Описание материальных потоков¹ с указанием вида, количества, периодичности поступления материалов, средств транспортировки и складирования. Для наглядного представления процессов преобразования поступающих в организацию материалов в готовую продукцию составляют схему материальных потоков. Схема вычерчивается в виде графа, узлы которого обозначают узлы хранения или переработки, а дуги передачу материалов. Другой системой описания материальных потоков является матрица (таблица) следующего вида:
- заголовки строк обозначают потребителей,
 - заголовки столбцов обозначают поставщиков,
 - в клетке указывается наименование, количество, периодичность поступления и другие характеристики потока материалов от поставщика потребителю.
5. Описание организационной структуры выполняют в виде схемы (обычно иерархического вида). Прямоугольники с соответствующими подписями (вершины) представляют подразделения (должностные лица), а соединяющие их линии – подчиненность, горизонтальные связи изображают пунктирной линией.
6. Функциональная схема увязывается с организационной и обычно включает:
- подразделение (должность),
 - функции, их значимость, входные и выходные данные, трудоемкость, явные и неявные правила выполнения.

Источниками перечисленных сведений являются учредительные документы и интервью с руководством организации. Важно, что описание давало полную картину о деятельности предприятия, но не было бы перегружено очевидными и излишними деталями, вроде описания функций бухгалтера материальной группы.

Кроме составления общего описания применяют специальные виды анализа.

Анализ критических факторов успеха

Критический фактор успеха – КФУ (Critical Success Factor) – это наиболее значимый фактор для решения некоторой задачи управления. КФУ

¹ Для управленческих структур (например, органов государственного управления) схема материальных потоков не приводится.

может определяться для каждой цели или задачи управления. Для определения проводится опрос менеджеров (ключевых пользователей). Достоинствами такого подхода является определение главных потребностей управления, широкий диапазон применения, легкость понимания. К недостаткам относятся сложности и противоречия в анализе и выборе КФУ и ориентация на потребности высшего руководства. Кроме КФУ рассматривают критические предположения, критические информационные потребности и критические показатели производительности. В результате анализа получается иерархическая система факторов. Для проектирования ИС эта система должна быть рассмотрена на предмет применения информационных технологий для решения проблем, обозначенных системой КФУ.

Анализ технологических цепочек

Данная методика заключается в изучении вклада в успех фирмы различных видов деятельности, которые классифицируют следующим образом:

1. Первичные виды деятельности – связанные с непосредственным созданием товаров и услуг, продаж и обслуживанием.
 - 1.1. Входная логистика – приобретение ресурсов производства.
 - 1.2. Производство – преобразование ресурсов в товары и услуги.
 - 1.3. Выходная логистика – предоставление товаров и услуг потребителям.
 - 1.4. Продажа и маркетинг товаров и услуг.
 - 1.5. Обслуживание.
2. Поддерживающие виды деятельности.
 - 2.1. Инфраструктура фирмы – технология учета, управление финансами, информационный сервис.
 - 2.2. Управление персоналом.
 - 2.3. Конструкторская и технологическая подготовка производства.
 - 2.4. Управление производством.
3. Прибыльность производства в виде системы показателей (маржа – разница между ценой единицы товара или услуги и стоимостью затрат на ее производство, чистый приведенный доход, точка безубыточности и другие).

SWOT-анализ

Следуя этой методике, выделяют сильные и слабые стороны предприятия, неиспользуемые возможности и возможные опасности. Результаты SWOT-анализа позволяют выделить наиболее перспективные сферы приложения информационных технологий.

В результате перечисленных видов анализа появляется список приоритетных проблем и возможностей организации. Далее выделяются такие

проблемы и возможности, для решения и использования которых необходимы информационные технологии. Информационные технологии могут применяться как для перестройки процессов управления (реинжиниринг бизнес-процессов), так и для поддержки существующих процессов управления.

5.2. Реинжиниринг бизнес – процессов

Современной технологией изменения бизнес – процессов является реинжиниринг бизнес - процессов [16].

Предшественником технологии реинжиниринга является "Трансформация и непрерывное усовершенствование бизнес-процессов" Эдвардса Деминга. Ее основными декларациями являются следующие:

- 1) Цель – повышение качества услуг, а не прибыль.
- 2) Трансформация организации работ, исходя из интересов потребителя.
- 3) Исследуются и устраняются недостатки производства, а не работника.
- 4) Внедряются командные (артельные) методы работы, в результате возрастает роль и ответственность каждого работника.

Появление реинжиниринга бизнес - процессов (Майкл Хаммер 1990) обусловлено качественными изменениями на рынке. Стремление получить конкурентные преимущества за счет более полного удовлетворения пользователя заставляет переходить от массового производства к индивидуальному даже в традиционно массовых отраслях, таких как производство мебели и автомобилей. При этом существенно возрастает нагрузка на управленческий аппарат и автоматизация управления становится необходимой частью реинжиниринга.

Реинжиниринг начинается с переосмысления положения фирмы на рынке. Однако с точки зрения построения ИС наиболее существенную роль играют изменения в технологии управления. Для этого применяются следующие преобразования:

- 1) Горизонтальное сжатие: несколько последовательно выполняемых рабочих процедур объединяют в одну.
- 2) Вертикальное сжатие: исполнители принимают самостоятельные решения по большому множеству задач, не обращаясь к вышестоящим руководителям.
- 3) Отказ от линейного упорядочения рабочих процедур в пользу их параллельного исполнения.
- 4) Процессы имеют разные варианты исполнения в зависимости от ситуации.
- 5) Работа выполняется в том месте, где целесообразно.
- 6) Уменьшается количество проверок и управляющих воздействий.
- 7) Минимизируется количество согласований.

8) Вводится "Уполномоченный менеджер" для принятия решений в рамках некоторого проекта.

В отличие от старых иерархических строго централизованных структур управления применяется централизованно-децентрализованный подход: происходит централизация хранения данных для их децентрализованного использования (в отличие от жестких иерархических структур, в которых "правая рука не знает, что делает левая").

Из приведенного выше описания становится понятно, что распределенная автоматизированная обработка данных становится необходимым компонентом реинжиниринга бизнес-процессов.

5.3. Планирование архитектуры информационной системы

На этом этапе выполняется описание и анализ существующей информационной системы организации, создается представление о будущей ИС. Составляется план ее построения. Определяются компьютерные технологии, хранимые данные, приложения, информационные услуги и технологическая инфраструктура будущей системы. Выделяются приоритетные подсистемы и приложения. Для этого выполняются следующие работы:

1. Описание модели ИС предприятия, в которую включают:
 - технические средства обработки информации;
 - топология сети;
 - применяемое программное обеспечение;
 - автоматизированные функции управления;
 - набор информационных услуг и порядок их предоставления;
 - систему хранимых данных;
 - географию узлов обработки данных.
2. Анализ существующих информационных услуг и технологий:
 - полнота автоматизации;
 - производительность системы;
 - степень интеграции различных подсистем;
 - соответствие современному уровню информационных технологий;
 - проблемы, возможности и ограничения информационной системы.
3. Выделение подсистем деловой активности с указанием приоритетов. Подсистема деловой активности выделяется при помощи логической группировки бизнес-процессов, их размещений, данных, которые поддерживаются несколькими связанными приложениями. Общим принципом такого выделения является минимизация внешних и максимизация внутренних связей. Для этого могут применяться различные формальные методы, например, кластер-анализ.
4. Разработка новой архитектуры ИС на основе выделенных подсистем. Архитектура ИС включает:

- принципы организации сети (топология и протоколы),
 - информационные потребности, технология хранения данных и обеспечение доступа к ним,
 - принципы обмена данными между приложениями,
 - специализация компьютеров в сети,
 - выбор операционной системы,
 - методы и технологии разработки программного обеспечения.
5. Планирование автоматизации подсистем деловой активности с учетом их актуальности, затрат на их реализацию и имеющихся ресурсов.
6. Обзор результатов и утверждение плана развития ИС.

Требовать и эффективности, и гибкости от одной и той же программы – все равно, что искать очаровательную и скромную жену... по-видимому, нам следует остановиться на чем-то одном из двух.

Джеральд Вейнберг.

6. Стадия системного анализа

Анализ выполняется для приложения или системы связанных приложений в рамках некоторого проекта. На основании описания и оценки существующей ИС, потребностей управления, проблем, новых возможностей для бизнеса, изменений в правилах ведения дел вырабатывается набор требований и приоритетов для приложений со стороны системы управления. Главный итоговый документ, разрабатываемый на этой стадии – это техническое задание.

На этой стадии можно выделить следующие этапы:

- Описание и анализ существующей технологии управления (в рамках проекта автоматизации).
- Определение требований и приоритетов для будущей системы (подсистемы или комплекса задач).

6.1. Описание и анализ существующей технологии управления

Для постановки задачи автоматизации управления, что является главной целью на этой стадии разработки, необходимо вникнуть в назначение и технологию процессов управления в рамках проекта. С этой целью составляется подробное описание бизнес-процессов.

Как и на стадии планирования строится описание организационно-функциональной структуры управления, выполняется схема материальных потоков (при необходимости), но выполняется это не для всей организации, а только для подразделений, затрагиваемых проектом. Кроме этого, приводятся схемы и описание регламента документооборота, строится ин-

формационная модель прикладной области, описывается география рабочих мест и действия, выполняемые на каждом рабочем месте.

Известно несколько видов диаграмм для описания документооборота. Для представления обмена информацией в укрупненной форме применяют схемы информационных потоков (см. рис. 9). Схемы являются ориентированными графами. Источники и получатели данных являются вершинами и изображаются овалами или прямоугольниками. Дуги соответствуют процессам передачи информации и нагружаются сведениями о передаваемых данных.

Для детального описания циркуляции документов используют схемы документооборота (см. рис. 10). Система обозначений для них предложена фирмой IBM и приведена в таблице 4.

Символы должны иметь стандартные пропорции (3×2) и одинаковые в рамках диаграммы размеры. Для удобства ссылок на компоненты схемы их можно снабжать номерами или другими обозначениями. Направления передачи документов указываются стрелками, кроме направлений сверху-вниз и справа-налево, которые считаются стандартными. Диаграммы потоков данных разделяют на полосы (или столбцы), указывая в одной полосе (столбце) операции, выполняемые в одном подразделении.

Основное отличие схемы документооборота от блок-схем является то, что линии обозначают передачу информации, а не управления. Для каждой операции должны быть указаны входные и выходные данные. Составление должно следовать этой логике обозначений. Например, недопустимо соединение двух операций линией транспортировки носителей без указания носителя, который передается от операции к операции. Появление документов, которые никем не используются (которые не имеют выходящих дуг), также не допустимо. Символ источника (получателя) информации используется для обозначения внешних, по отношению к рассматриваемой предметной области, объектов, детализация документооборота внутри которых не нужна.

Схемы информационных потоков и документооборота дополняют пояснениями, образцами форм документов, правилами заполнения и обработки документов, временным регламентом обработки документов (сроками поступления документов и временем выполнения операций).

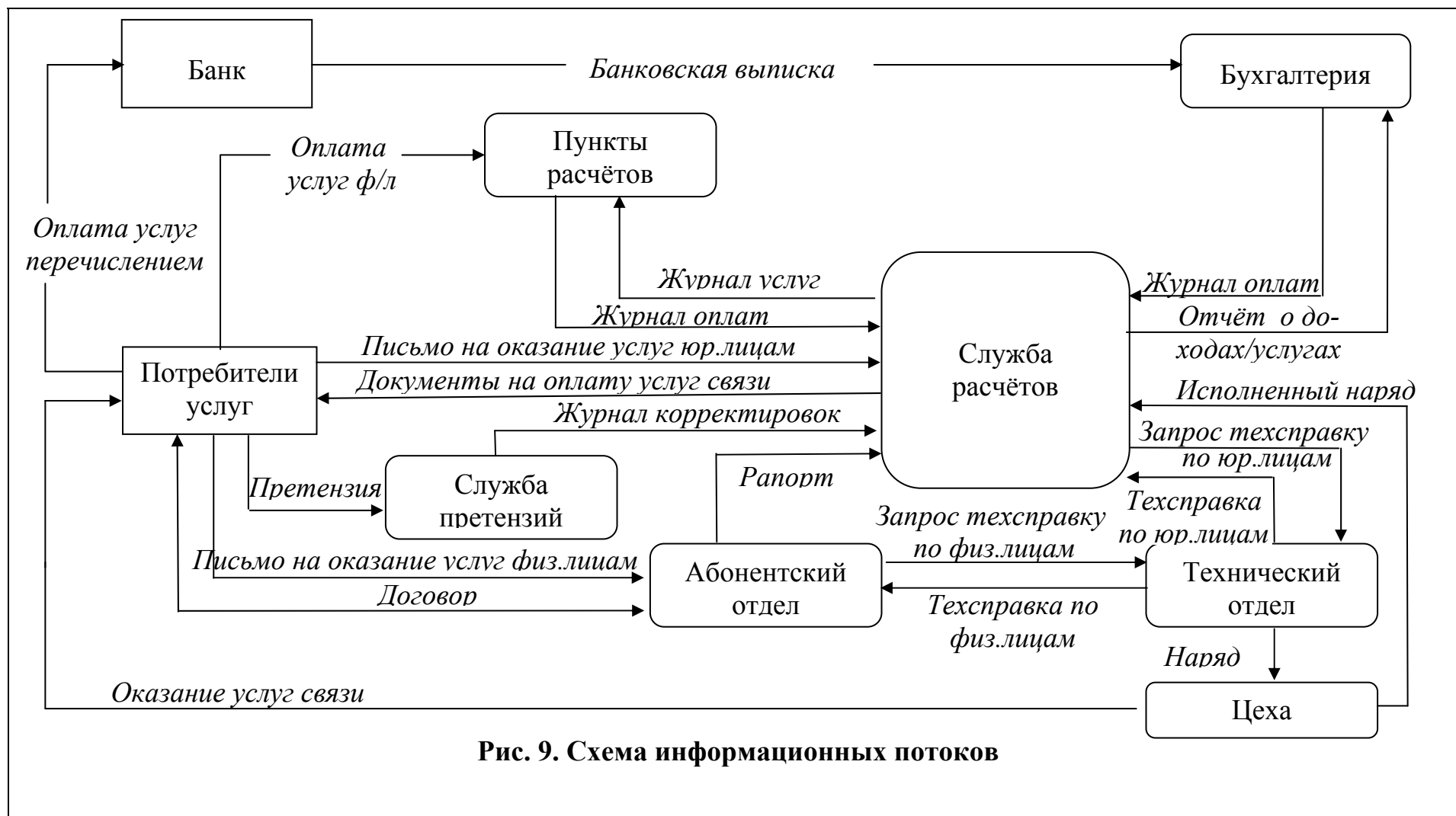


Рис. 9. Схема информационных потоков

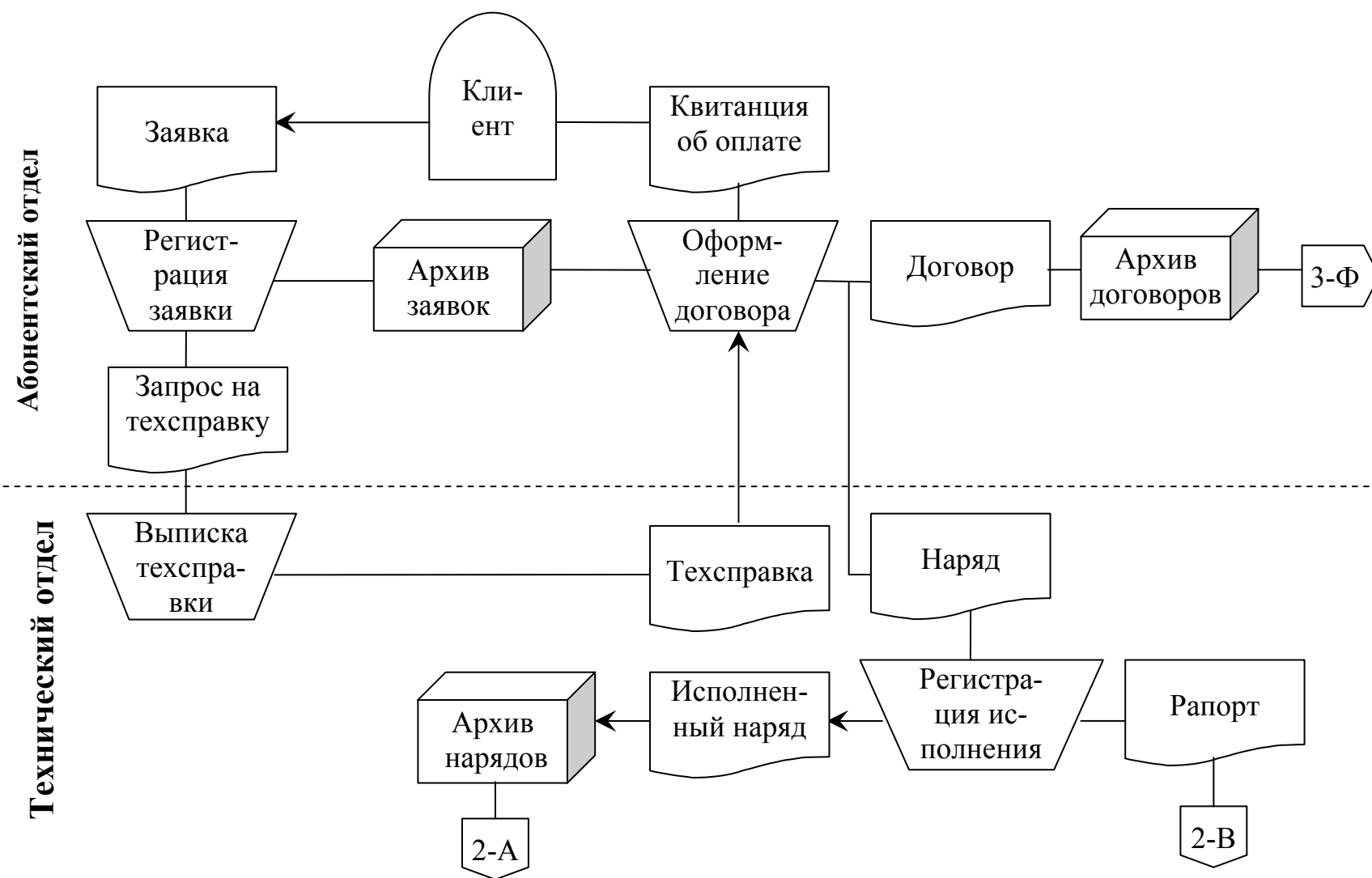
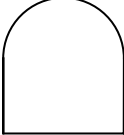
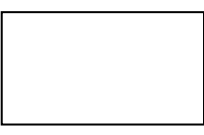
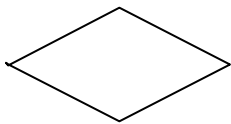
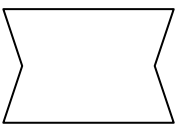
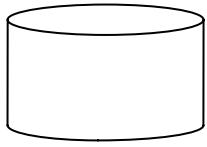
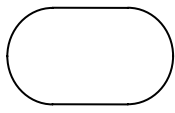
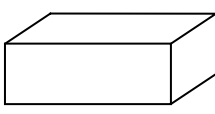
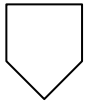
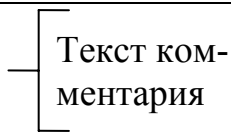
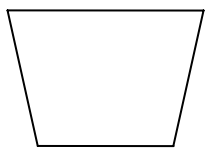
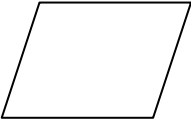
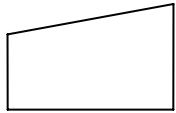
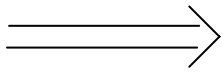
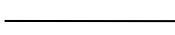


Рис. 10. Пример схемы документооборота

Таблица 4

Символы схем описания документооборота

Символ	Пояснения	Символ	Пояснения
	источник-получатель информации		операция, выполняемая на ЭВМ
	развилка		ручной документ
	магнитный диск		пуск – останов
	напечатанный ЭВМ документ		внутристраничный соединитель
	архив		межстраничный соединитель
	неавтономная память		комментарий
	ручная операция		ВВОД-ВЫВОД
	ввод с клавиатуры		материальный поток
	канал связи		транспортировка носителей

Информационная модель организации – это описание содержания структуры и источников данных, используемых в управлении обычно в виде структурной информационной модели (см. п.7.4.1) Отправной точкой

построения служат входные и выходные документы, исходные и выходные данные функций.

Инструментарий информационной модели позволяет описать основные информационные блоки в виде набора документов, объединяющих показатели и другие единицы информации. На этапе анализа можно ограничиться крупными информационными единицами, не завершая детализацию элементарными атрибутами.

Все эти сведения являются основой для анализа существующей технологии управления. Могут применяться технологии исследования, используемые на стадии планирования (построение системы критических факторов успеха, SWOT-анализ, реинжиниринг бизнес-процессов), только в масштабах не организации, а сферы управления, затрагиваемой проектом. Еще раз следует подчеркнуть, что автоматизация должна приводить к инновации процессов управления на основе информационных технологий.

Анализ технологии управления включает идентификацию проблем управления, упускаемых возможностей и изменений в условиях выполнения бизнес-процессов, а также определение целесообразности решения каждой проблемы, использования каждой возможности, необходимости изменений информационной системы для соответствия новым условиям.

Анализ проблем и возможностей может состоять из следующих пунктов:

- анализ производительности (объем работы (транзакций) в единицу времени, время выполнения работы),
- анализ предоставляемых (выходных) данных по направлениям:
 - недостаток информации для решения,
 - форма информации неудобна для управления,
 - несвоевременность информации,
 - избыток информации,
 - неточность информации,
- анализ хранимых данных – выявление следующих свойств:
 - полнота,
 - целостность,
 - избыточность,
 - гибкость (возможность выполнения сложных запросов),
- экономический анализ (стоимость и прибыльность проектных решений),
- анализ управляемости (много или мало управления) и безопасности,
- анализ эффективности (возможности увеличить и/или уменьшить производительность),
- анализ сервиса:
 - точность,

- надежность,
- простота использования,
- гибкость (управление исключительными, нестандартными ситуациями),
- координированность.

На основании анализа предлагаются цели и ограничения новой системы. Цели формулируются как показатели успеха в решении проблем и использовании возможностей. Ограничения устанавливают границы (временные, стоимостные, технологические, политические), в которых ищется решение.

6.2. Определение требований и приоритетов

На данном этапе определяются требования к информационной системе. Приоритетными являются функциональные требования: что и как должно быть автоматизировано. Кроме этого, формулируются технологические требования по всем видам обеспечения. Требования формируются с учетом трудовых и временных затрат, а также других ресурсов, необходимых для реализации проекта. Учитываются совместимость с существующей информационной системой и перспективы развития. В результате создается и утверждается принципиально важный документ – техническое задание на проектные работы.

Техническое задание регламентирует взаимоотношения заказчика и исполнителя работ, поэтому кроме требований оно содержит план выполнения работ, размеры и порядок финансирования.

Для формирования технического задания выполняются следующие работы:

1. Идентификация требований.
2. Назначение приоритетов требованиям: обязательные, желательные (реализуемые позже), по выбору.
3. Построение плана и бюджета проекта.
4. Оформление и утверждение технического задания.

6.3. Содержание технического задания

Существует ГОСТ 34.602-89, регламентирующий содержание и оформление технического задания на создание автоматизированной системы. По причине важности этого документа ниже приводятся основные положения ГОСТа.

Техническое задание включает следующие пункты:

- 1) Общие сведения:
 - наименование системы и ее условное обозначение;
 - номер договора или шифр темы;
 - наименования и реквизиты предприятий разработчика и заказчика;

- перечень документов, на основании которых создается система;
 - сроки выполнения работ;
 - сведения об источниках и порядке финансирования;
 - порядок предъявления результатов.
- 2) Назначение и цели создания (развития) системы:
- назначением являются виды автоматизируемой деятельности, дополненные перечнем объектов автоматизации;
 - цели формулируют как требуемые значения показателей, которые должны быть достигнуты в результате внедрения системы.
- 3) Характеристики объекта автоматизации:
- краткие сведения об объекте или ссылка на соответствующие документы;
 - условия эксплуатации системы на объектах автоматизации.
- 4) Требования к системе:
- требования к системе в целом:
 - к структуре и функционированию системы: компоненты системы, способы и средства связи компонент; взаимосвязи с другими системами;
 - к режимам функционирования; к диагностическим процедурам;
 - перспективы развития и автоматизации;
 - к численности, квалификации и режиму работы персонала;
 - к надежности: показатели надежности; перечень аварийных ситуаций; требования к надежности технических средств и программного обеспечения, требования к методам оценки надежности;
 - к эргономике и технической эстетике;
 - требования к защите информации от несанкционированного доступа;
 - к сохранности информации при авариях;
 - Требования к функциям (задачам):
 - перечень функций (задач, комплексов);
 - регламент реализации каждой функции;
 - для каждой функции требования к качеству, форме представления выходных данных, точности, ресурсам для выполнения.
 - Требования к видам обеспечения:
 - к математическому обеспечению: состав алгоритмов, типовые алгоритмы, требования к алгоритмам;
 - к информационному обеспечению:
 - к составу, структуре и способам организации данных;
 - к информационному обмену между компонентами системы;
 - к информационной совместимости с другими системами;

- к использованию классификаторов;
 - к применению СУБД;
 - к процессу сбора, обработки, передачи и представления данных;
 - к защите от разрушения при авариях и сбоях;
 - к контролю хранения, к обновлению и восстановлению данных;
 - к процедуре придания юридической силы документам, полученных при помощи технических средств системы;
 - к программному обеспечению: требования к независимости программ от вычислительной техники и операционной системы; к качеству, способам обеспечения качества, к контролю качества;
 - к техническому обеспечению: к видам и характеристикам технических средств;
 - к организационному обеспечению: к структуре и функциям подразделений, участвующих в функционировании и обеспечивающих эксплуатацию; к организации функционирования и порядку взаимодействия персонала АС и персонала объекта автоматизации; к защите от неправильных действий персонала.
- 5) Состав и содержание работ по созданию системы:
- перечень стадий и этапов разработки, сроки выполнения, исполнители;
 - перечень документов, предъявляемых по окончании каждого этапа работ.
- 6) Порядок контроля и приемки системы:
- виды, состав, объем и методы испытаний;
 - общие требования к приемке работ, порядок согласования и утверждения приемочной документации;
 - состав приемочной комиссии.
- 7) Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие:
- приведение поступающей в систему информации к виду, пригодному для обработки;
 - изменения в объекте автоматизации;
 - создание условий функционирования объекта автоматизации для соответствия условия эксплуатации системы;
 - создание подразделений и служб, необходимых для функционирования системы;
 - сроки, порядок комплектации штатов и обучения персонала.
- 8) Требования к документированию – перечень комплектов документации на бумажных и машинных носителях.

9) Источники разработки – документы, на основании которых разрабатывается ИС.

При наличии утвержденных методик в техническое задание включают расчет ожидаемой эффективности системы; оценку научно-технического уровня системы.

Оформление титульного листа технического задания приведено на рис. 11. Листы нумеруют, начиная с листа, следующего за титульным. Заключительный лист включает две приведенные ниже таблицы с подписями.

СОСТАВИЛИ

Наименование предприятия	Должность исполнителя	Фамилия имя, отчество	Подпись	Дата

СОГЛАСОВАНО

Наименование предприятия	Должность исполнителя	Фамилия имя, отчество	Подпись	Дата

УТВЕРЖДАЮ

<руководитель (должность, наименование предприятия – заказчика)>
 <личная подпись> <расшифровка>
 <Печать> <дата>

УТВЕРЖДАЮ

<руководитель (должность, наименование предприятия – разработчика)>
 <личная подпись> <расшифровка>
 <Печать> <дата>

<наименование системы (проекта)>

<наименование объекта автоматизации>

<сокращенное наименование автоматизированной системы>

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

На <количество листов> листах

Действует с <дата>

СОГЛАСОВАНО

<руководитель (должность, наименование согласующей организации)>
 <личная подпись> <расшифровка>
 <Печать> <дата>

Рис. 11. Оформление титульного листа технического задания

Стандарт предлагает полный набор пунктов технического задания. На практике незначимые для конкретного проекта пункты опускают. Например, если организация выполняет проект силами своих работников, то из

пункта «Общие сведения» исчезают номер договора или шифр темы, реквизиты разработчика.

Исполнитель представляет проект технического задания на согласование и утверждение. Согласование проекта не должно превышать 15 дней. Замечания по проекту технического задания представляются с техническим обоснованием. Разногласия заказчика и разработчика оформляются протоколом. Копии технического задания в 10-дневный срок рассылаются исполнителям.

В характеристике объекта автоматизации приводят только те сведения, которые значимы для разработки: характеристики сети и компьютеров, используемые операционные системы, квалификация персонала и т.д.

Наиболее важный пункт – «Требования к системе». В требованиях к системе в целом главным является структура проектируемой системы. Перечисляют компоненты системы – обычно ее составляют серверы (такие как серверы баз данных) и клиентские приложения. Для каждой компоненты приводят функции и описывают регламент взаимодействия компонент. Функции каждого компонента должны быть более подробно описаны в п. «Требования к функциям».

Как правило, алгоритмы функций системы общеизвестны. Это могут быть, например, функции учета или определение финансовых показателей. Описание таких функций не включается в требования к математическому обеспечению.

Простое перечисление стандартных характеристик операционных систем и вычислительной техники обычно составляет содержание пунктов «требования к программному обеспечению» и «требования к техническому обеспечению».

Требования к информационному обеспечению в случае использования СУБД сводятся к описанию ее характеристик, технологии связи с другими серверами, процесса редактирования данных, защиты данных.

Требования к документированию содержат список документов, который можно разделить на две части: инструкции по использованию и сопровождению и описание информационных и программных продуктов. Все прочие пункты технического задания стандартны и их разработка не вызывает затруднений.

Большой программистский проект ... включает в себя много отдельных задач, каждая из которых может зависеть от окончания другой. Вероятность того, что каждая задача будет идти нормально, становится исчезающе малой.

Фредерик Брукс-мл.

7. Конструирование

7.1. Работы стадии конструирования

На этой стадии предстоит разработать архитектуру – каркас будущей информационной системы. На основании требований к системе с учетом мнений и рекомендаций пользователей вырабатываются принципиальные информационные решения в части структур хранения данных, методов и алгоритмов вычислений, структуры сети, интерфейса пользователей. Однако, прежде надо решить стоит ли разрабатывать программы и структуры данных или лучше воспользоваться готовыми. Второй вариант, безусловно, намного проще и надежнее, если на ранке есть продукт с нужным набором функциональных характеристик. Покупка выполняется обычно по следующей схеме:

- 1) Формулировка требований.
- 2) Сбор предложений поставщиков.
- 3) Проверка надежности и исполнительности поставщиков.
- 4) Оценка и ранжирование предложений.
- 5) Разработка контрактов.

Если все же выбирается вариант разработки, то в этом случае и нужно выполнять конструирование и реализацию системы. Конструирование включает следующие виды работ:

1. Построение функциональной модели системы. Выполняется декомпозиция функций системы для получения некоторого элементарного уровня. Для каждой функции описываются входные и выходные данные, алгоритмы выполнения, взаимодействие с другими функциями, регламент выполнения функции.
2. Построение модели данных. Определяется, какие данные надо хранить для выполнения функций информационной системы. Кроме этого, должны учитываться все информационные потребности организации и перспективы развития информационной системы. После этого нужно решить вопросы о форме хранения данных на основании требований к доступу, надежности, безопасности данных.
3. Определение набора приложений. Обычно приложения определяются количеством и функциональными характеристиками рабочих мест.

4. Проектирование функций, входных и выходных данных каждого приложения. Одна и та же функция может применяться в разных приложениях. В таких случаях следует определить модульную или компонентную структуру всего проекта.
5. Проектирование интерфейса пользователя. Главная проблема здесь заключается в единообразном применении стандартной терминологии и компонентов интерфейса.
6. Проектирование структуры и логики работы программы.

На стадии конструирования особенно целесообразно использование единого для всех разработчиков репозитория для хранения документации на подсистему и всех описаний, связанных с проектом. Применение репозитория решает задачи распространения изменений в документации и заменяет передачу данных от этапа к этапу предоставлением их по мере готовности, что создает предпосылки для распараллеливания и децентрализации работ.

Конструктивные решения оформляются набором документов. Ниже приведен список документов, разработку которых предусматривают ГОСТы и которые целесообразно разрабатывать на стадии конструирования.

- Пояснительная записка к эскизному (техническому) проекту.
- Общее описание системы.
- Схема организационной структуры.
- Описание организационной структуры.
- Схема функциональной структуры.
- Описание автоматизируемых функций.
- Описание постановки задачи (комплекса задач).
- Описание технологического процесса обработки данных.
- Схема автоматизации.
- Описание комплекса технических средств.
- Описание информационного обеспечения.
- Описание организации информационной базы.
- Описание алгоритма (проектной процедуры).

Требования к содержанию этих документов приведены в приложении.

Известно, что программисты не любят писать документацию. Среди них даже бытует мнение, что это форма наказания для неудобных. Есть одно оправдание такой ситуации: много документов создается раньше программ и они теряют актуальность по мере внесения изменений в проект на стадии реализации. Таким образом, создается впечатление, что документы оформлены зря, так как их все равно приходится переделывать. Тем не менее, все разработчики, а особенно руководители проектов должны знать,

что если разработка не документируется, то риск неудачи существенно возрастает.

С другой стороны, изменения в технологии проектирования требуют новых подходов к документированию. Наверное, никто уже не вычерчивает эскизы окон, поскольку все компоненты интерфейса пользователя просто размещаются на экране монитора при помощи графических инструментальных средств проектирования. В данном случае описание непосредственно становится программой. Такой «симбиоз» описаний и программ является идеальным вариантом документирования проекта.

К сожалению, значительная часть проектных решений не имеет жестких связей со структурами программ и данных. Это является основным препятствием создания конверторов из описаний в программы и наоборот. К тому же для такого конвертирования требуется создание формальных языков, которые обычно плохо воспринимаются неподготовленным пользователем. Тем не менее, постоянно предпринимаются усилия автоматизировать сам процесс проектирования и документирования информационной системы. Результатом таких усилий являются различные CASE - средства.

7.2. CASE-технология проектирования информационных систем

CASE – это акроним Computer-Aided System / Software Engineering, объединяющий методы автоматизации самого процесса проектирования информационных систем [10, 13, 18, 21]. CASE средства поддерживают или автоматизируют работы на всех этапах жизненного цикла системы. Целью CASE-технологии является ускорение процесса создания систем и повышение их качества.

С начала по середину 70-х Daniel Teichrowe создает Problem Statement Language (PSL) и для него разрабатывают программу Problem Statement Analyzer (PSA), выполняющую анализ требований пользователя на полноту и последовательность. В 1984 компания INTERSOLV (прежнее название IndexTechnology) выпустила Exelerator. Его успех породил термин CASE и индустрию CASE-средств.

По степени охвата этапов жизненного цикла CASE-программы классифицируют следующим образом:

- Upper-CASE для поддержки планирования, анализа и общего конструирования систем,
- Lower-CASE для детального конструирования, разработки и сопровождения,
- Cross life cycle-CASE для поддержки систем на всех этапах жизненного цикла.

В состав Upper-CASE включают:

- Средства планирования развития информационных систем, автоматизирующие описание целей и задач, определение критических факторов успеха, формулирование проблем, описание организационной структуры, информационных и функциональных требований.
- Средства системного анализа и конструирования, позволяющие описать границы проектируемой системы, модель существующей системы, выполнить конструирование системы, отвечающей требованиям пользователей, создать прототипы специфических частей системы (отчетов, экранных форм, ...).

В состав Lower-CASE включают

- Средства конструирования и реализации систем, которые делят на три категории:
 - инструментальные средства прикладного программиста, поддерживающие кодирование, компиляцию, тестирование и отладку на интеллектуальных терминалах,
 - генераторы компонент, выполняющие генерацию соответствующих компонент по их описанию (генераторы БД, создание графического интерфейса пользователя, средства реализации технологии "клиент-сервер"),
 - генераторы программ (кода) создающие текст программ по описаниям системы.
- Средства поддержки ИС:
 - помогают программистам реструктурировать текст старой программы, чтобы сделать её более мобильной,
 - помогают изменять систему по требованиям пользователя,
 - облегчают переход на новые технологии,
 - помогают определить стоит ли еще поддерживать ИС или необходимо создавать её заново,
 - восстанавливают полезную информацию из устаревшего программного обеспечения.

Главным в средствах поддержки является проектирование новых программ, а также изменение и реструктуризация существующих программ в соответствии с описаниями.

Противоположные средства – reverse engineering (design recovery) – восстанавливают описания более высокого уровня по описаниям низкого уровня (по выполняемому модулю восстанавливается текст программы, по программе – структурные схемы, диаграммы потоков, деревья решений). Программа, как конечный продукт проектирования, содержит не только идеи программиста, но также и идеи системного аналитика и требования пользователей. Все это может быть использовано в новой системе. При этом следует учитывать, что старые программы были написаны без соблю-

дения стандартов и извлекаемая в таких случаях информация формулируется в физических терминах (например в адресах переменных).

В Cross life cycle CASE включают следующие средства:

- управление проектом – планирование, составление расписаний и бюджетов, составление отчетов, управление ресурсами;
- управление процессами;
- оценивание времени и стоимости проекта, а также прочих ресурсов;
- документирование.

Архитектура CASE-средств продемонстрирована на рис. 12. *Репозиторий* (энциклопедия, словарь данных) является ядром системы и представляет собой специализированную базу данных, на основе которой выполняется синхронизация всех описаний (диаграмм) и координация усилий всех разработчиков. Репозиторий содержит информацию об объектах проектируемой ИС и взаимосвязях между ними:

- проектировщики и их права доступа к различным компонентам системы;
- организационные структуры;
- диаграммы, отражающие организационные, функциональные, информационные, технологические и другие аспекты системы;
- компоненты диаграмм;
- связи между диаграммами;
- структуры данных;
- программные модули;
- процедуры;
- библиотеки модулей и т.д.

Графический редактор диаграмм позволяет создавать различные описания системы в виде рисунков и может выполнять следующие операции:

- создавать элементы диаграмм и взаимосвязи между ними;
- задавать описания элементов диаграмм;
- задавать описания связей между элементами диаграмм;
- редактировать элементы диаграмм, их взаимосвязи и описания.

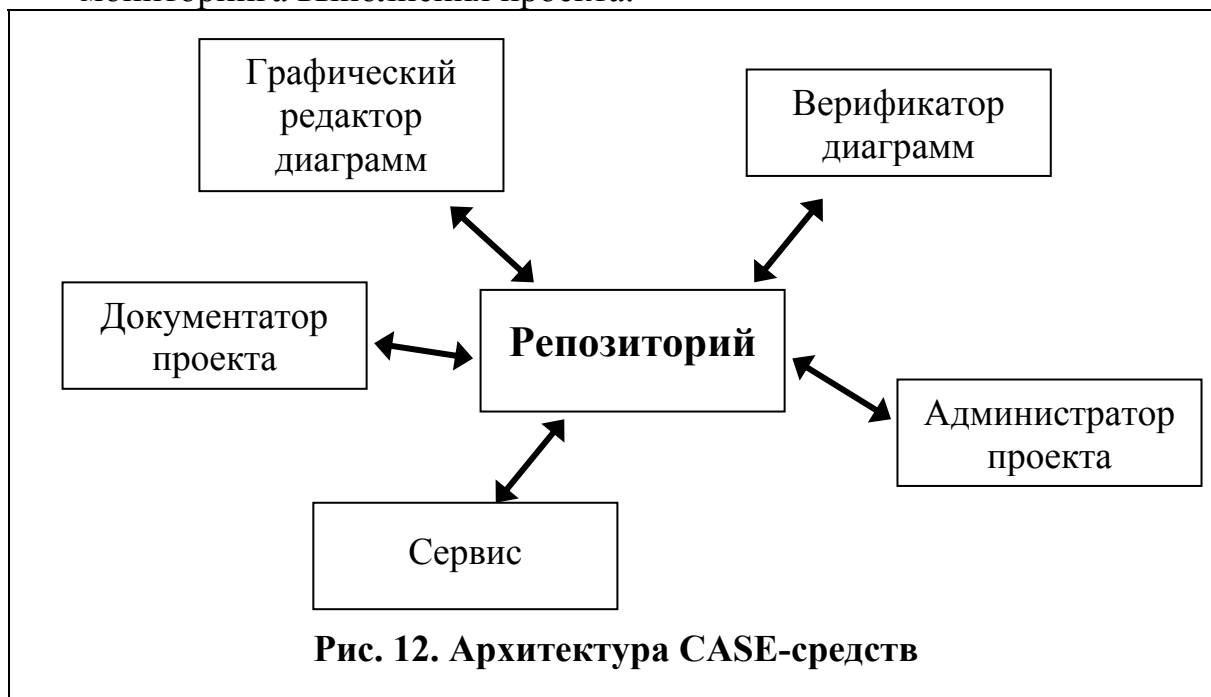
Верификатор диаграмм контролирует правильность диаграмм и синхронизирует диаграммы между собой.

Документатор проекта документирует проект в виде различных отчетов. Отчеты могут строиться по нескольким признакам, например, по времени, автору, элементам диаграмм, диаграмме или проекту в целом.

Администратор проекта автоматизирует выполнение следующих административных функций:

- инициализации проекта;
- задания начальных параметров проекта;

- назначения и изменения прав доступа к элементам проекта;
- мониторинга выполнения проекта.



Сервис выполняет функции архивации данных, восстановления данных и создания нового репозитория.

Основные возможности CASE-средства определяются набором моделей, который поддерживает графический редактор диаграмм. Некоторые такие модели, используемые на стадиях планирования и анализа (организационная, функциональная, схемы документооборота,...), уже были рассмотрены. Модели, применяемые на стадии конструирования, образуют метафору проектирования или подход к проектированию. Выделяют три подхода к проектированию [17]:

- Функционально-ориентированное проектирование, которое использует структурные методы для построения функциональной, информационной и других моделей информационной системы.
- Объектно-ориентированное проектирование [5] предлагает набор объектных моделей для описания предметной области. Для этого применяется специальный язык UML (Unified Modeling Language) [6].
- Модельно-ориентированное проектирование основано на настройке и доработке типовой конфигурации информационной системы в среде специализированных инструментальных систем, таких как R/3, BAAN, 1C:Предприятие.

7.3. Построение функциональной модели информационной системы

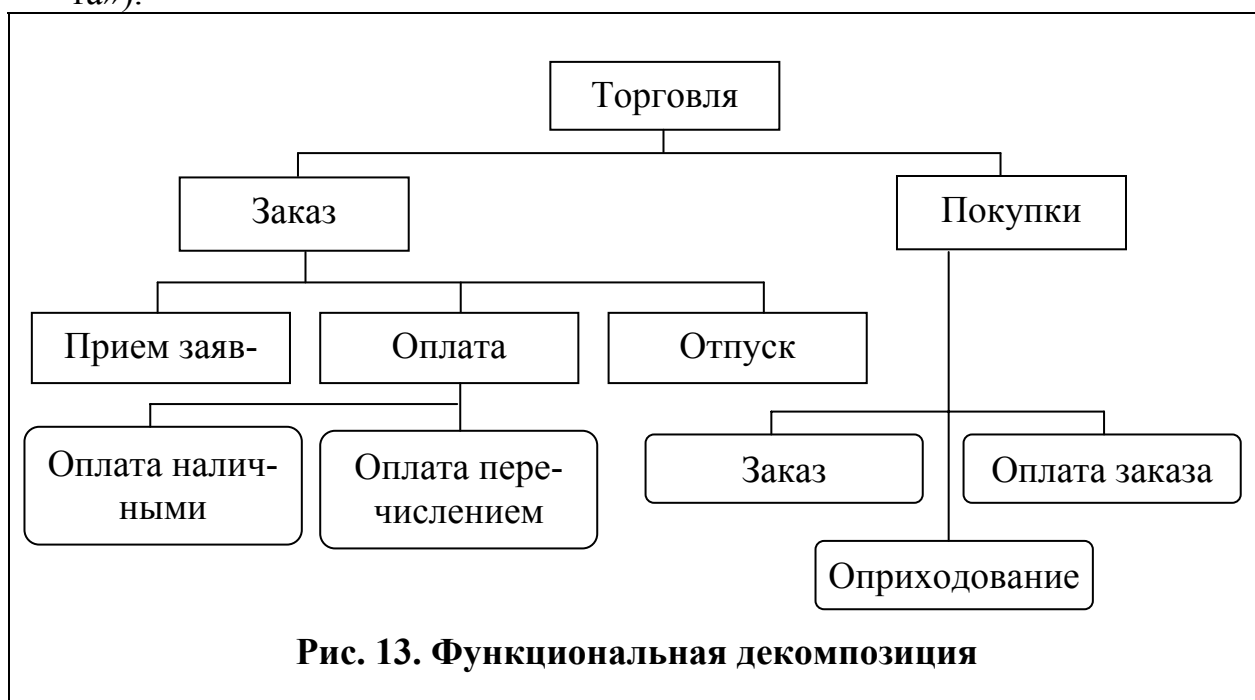
Методологией построения функциональной модели является декомпозиция, в результате которой получается иерархическая система функций.

Иерархические системы функций изображают функциональными диаграммами (см. рис. 13). Разбиение выполняется до уровня элементарных функций (прямоугольники со скругленными углами на рис. 13), которые обладают следующими свойствами:

1. Элементарная функция выполняется без прерываний, т.е. в процессе своего выполнения не ждет наступления некоторого события.
2. Время выполнения элементарной функции не зависит от выполнения других процессов:
 - известен вход функции: он доступен из внешней среды или хранится внутри системы,
 - функция не ждет востребования результатов решения: выход передается во внешнюю среду, или другому процессу, или хранится внутри системы.
3. Процесс является однородным с точки зрения пользователя (один и тот же тип входных данных, обработки и выходных данных).

В процессе структурирования используют два типа разбиения функции на составляющие:

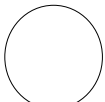
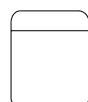
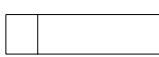
- дифференциация разбивает функцию на функции, выполняемые последовательно (см. разбиение функции «Заказ» на рис. 13),
- специализация расщепляет функцию на составляющие функции, требующие разной технологии исполнения (см. разбиение функции «Оплата»).



Для более детального описания функций необходимо указать входные и выходные данные для каждой функции и описать порядок выполнения и правила взаимодействия функций. Для наглядного изображения связи

функций, данных и пользователей применяют диаграммы потоков данных (Data Flow Diagram – DFD). Для построения диаграмм используются две системы обозначений, символы которых приведены в таблице 5. Символы соединяются линиями со стрелками, изображающими потоки данных.

Таблица 5
Символы диаграмм потоков данных

Система Yordon/DeMacro		Система Gane & Sarson	
Символы	Пояснения	Символы	Пояснения
	– Процесс		– Процесс
	– Файл		– Хранилище данных
	– Источник-приёмник данных		– Внешний объект

Построение диаграмм выполняется по следующим правилам:

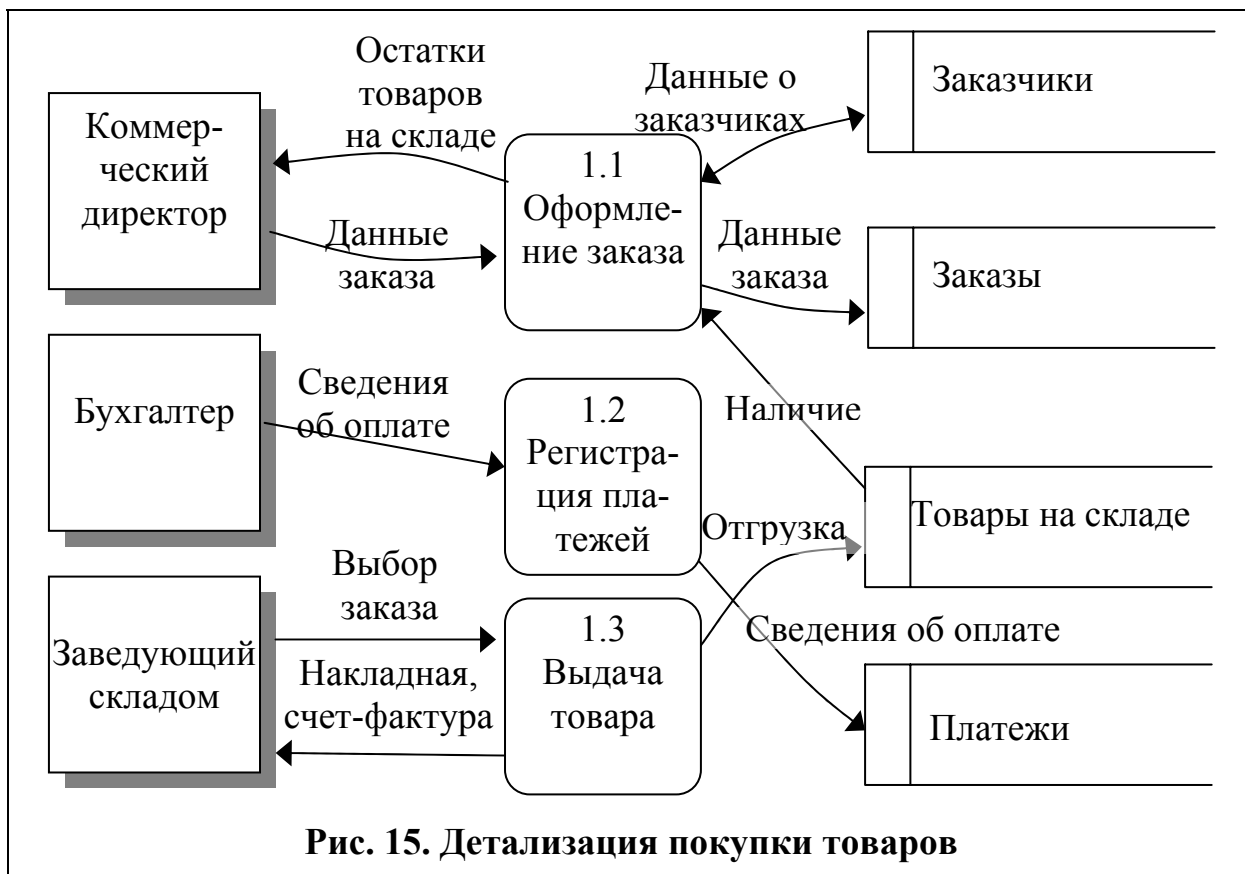
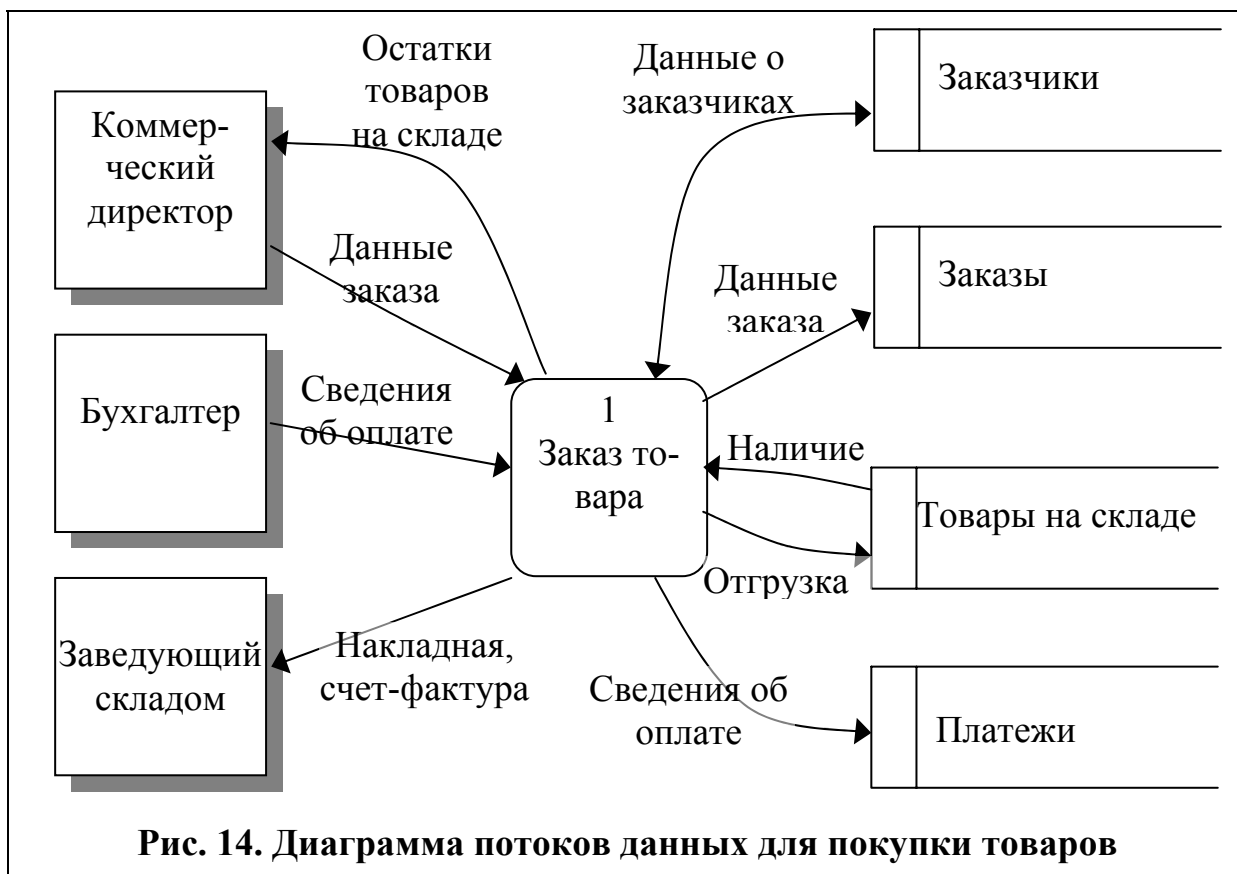
1. Компоненты соединяют стрелками попарно. Стрелка обозначает передачу данных. Стрелку помечают списком передаваемых данных.
2. Два элемента соединяют одной стрелкой, даже если она будет обозначать передачу нескольких данных.
3. Неправильно соединять стрелкой два объекта, два хранилища данных, объект и хранилище.
4. Процесс обязательно должен иметь входные и выходные данные.

На рис. 14 приведена диаграмма потоков данных оптовой покупки товара.

Преимуществом такого вида диаграмм (по сравнению, например со схемами документооборота) является то, что они позволяют отображать пошаговую детализацию. Декомпозицию задачи «Формирование заказа» представлена на рис. 15. Для удобства все процессы нумеруются. При декомпозиции процесса на составляющие им присваивается номер, который содержит обозначение исходного процесса и номер компоненты, разделенные точкой.

Диаграммы потоков данных сопровождают описаниями, которые содержат следующие сведения:

1. Определение ручных и компьютерных процессов.
2. Определение диалоговых и пакетных процедур.
3. Определение циклов (дневные, недельные,...) и событий, которые их инициируют (например, обращение клиента за обслуживанием).
4. Определение места выполнения процессов (сетевые узлы).



5. Распределение хранилищ данных по узлам сети.
6. Распределение процессов по узлам.
7. Выбор технологии реализации.

Кроме диаграмм функциональной декомпозиции и потоков данных применяют и другие системы обозначений:

- IDEF0 для описания технологических цепочек (для каждой функции указывают вход, выход, управление, механизмы (ресурсы) в виде линий, связывающих данную функцию с другими),
- IDEF3 (workflow diagramming) – описание логики взаимодействия информационных потоков и объектов, являющихся частью этих процессов.

Диаграммы потоков данных, более других видов диаграмм, подходят для описания функционирования информационных систем. В частности, они определяют какие данные циркулируют и накапливаются в системе. Эти сведения необходимы для построения моделей данных.

7.4. Построение модели данных

Модели данных применяют для решения различных задач и это привело к значительному количеству систем обозначений и методик построения моделей. Для понимания информационного описания предметной области и построения структур файлов применяют структурные модели данных. Для организации эффективного накопления и использования данных в многопользовательской среде применяют модели, ориентированные на регулярные структуры, применяемые в базах данных. Попытки внести в модели данных правила обработки данных (так называемые бизнес-правила) привели к созданию баз знаний и соответствующих инструментов построения новых выводов на основании данных и базовых правил.

7.4.1. Структурированное описание данных

Структурированное описание данных [15] названо так не потому, что все другие модели являются неструктурными, а потому, что оно более других методов связано с пошаговой детализацией, характерной для всех структурных методов. Основу структурированного описания данных составляют элементарные единицы информации и правила образования составных единиц информации (СЕИ).

Элементарной единицей информации (или атрибутом, или реквизитом) называют неделимую в смысловом отношении единицу информации. Единица информации (ЕИ) характеризуется именем, типом и значением. Типы элементарных ЕИ аналогичны элементарным типам данных в языках программирования – это целые и вещественные числа, логические данные, строки постоянной или переменной длины, даты и появившееся относи-

тельно недавно Binary large object (BLOB) – структуры произвольной величины для хранения графики, текстов и вообще любых объектов.

Составные единицы образуются либо как структуры – коллекция произвольных единиц информации разных типов – полей, выбираемых по имени, либо как массивы – набор однотипных единиц информации, выбираемых по номеру. Динамические структуры: списки, деревья и другие – используются в структурированных моделях данных значительно реже.

При наличии сложных структур и зависимостей между ними информационную модель отображают в виде диаграммы (см. рис. 16). Атрибуты изображаются прямоугольниками, структуры – в виде дерева: корень – название структуры, потомки – поля. Массив изображается на схемах прямоугольником, наложенным на прямоугольник. Условно присутствующие ЕИ- изображаются на схемах прямоугольником, нарисованным пунктиром.

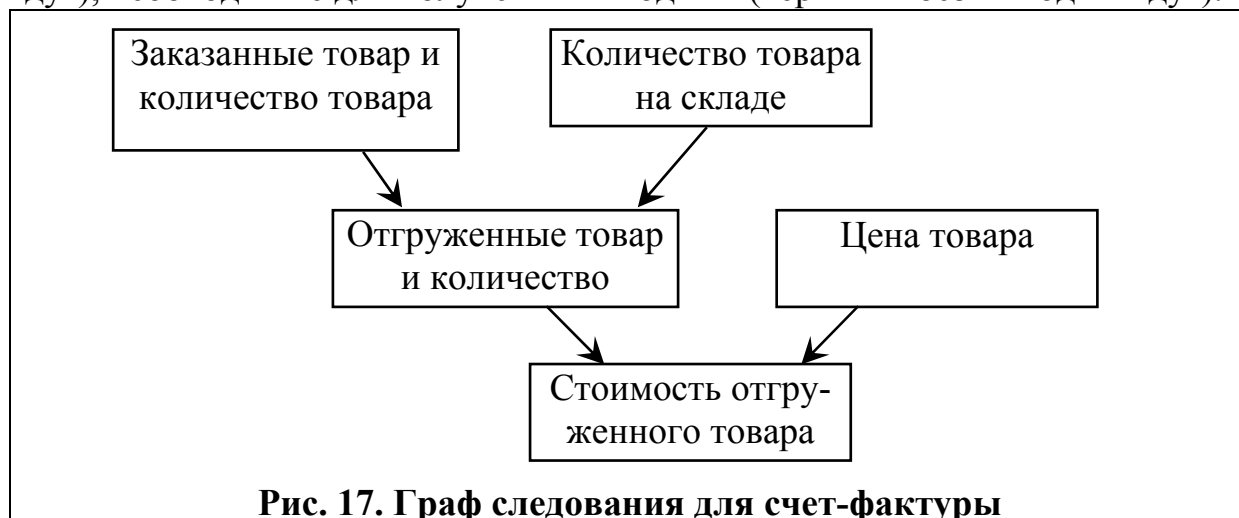


Все атрибуты делят на атрибуты-признаки и атрибуты-основания. Атрибут-признак отображает качественную характеристику, а атрибут-основание – количественную. Экономический показатель как некоторая количественная характеристика экономического объекта или процесса описывается некоторым набором атрибутов-признаков и одним атрибутом основанием. Атрибуты-признаки указывают какую именно характеристику определяет атрибут-основание. Например, величина прибыли (число) должна быть дополнена признаками, описывающими период, за который прибыль была получена, фирмы, которая прибыль получила, и так далее.

Информационная модель, использующая понятие показателя, имеет вид набора документов, каждый из которых состоит из определенного набора показателей.

Для оценки полноты информационной модели строят граф следования (предшествования) (см. рис. 17). ЕИ₁ следует из (предшествует) ЕИ₂, если ЕИ₁ получается на основе (используется для формирования) ЕИ₂. Граф

следования позволяет определить исходные ЕИ (вершины без входных дуг), необходимые для получения выходных (вершины без выходных дуг).



Структурные модели данных полезны для описания алгоритмов и структур данных в программах. Иногда на их основе определяют структуры файлов. Однако, специфические структуры файлов редко применяют для построения экономических информационных систем. Обычно, информация хранится в виде баз данных. Это позволяет использовать готовые решения и программы по накоплению и обработке данных.

7.4.2. Модели данных для СУБД

Модели данных для СУБД должны предоставлять эффективные средства для хранения, доступа и выборки данных. Базы данных изначально предназначались для использования в многопользовательской среде и обеспечения независимости данных и программ [8].

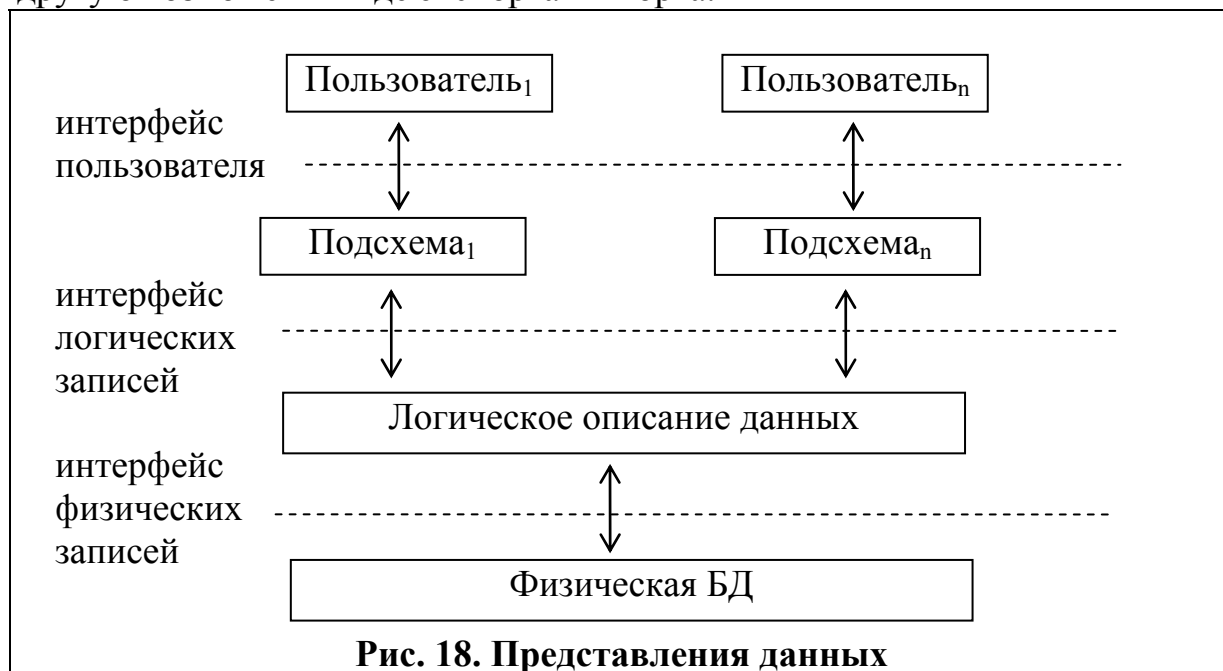
Выделяют три представления данных в СУБД (см. рис. 18):

- Представление пользователя (или внешняя модель или подсхема).
- Логическое описание (или схема или логическая БД).
- Внутреннее описание (или физическая БД).

Перечисленные модели связаны друг с другом соответствующим интерфейсом (реализованным при помощи СУБД). Выделение этих моделей позволяет свести изменения в программном и информационном обеспечении при изменении представлений пользователей, логического описания данных или физической БД к изменению соответствующих интерфейсов. Это и обеспечивает независимость программ и данных.

Для построения многопользовательских систем используют, в основном, SQL-серверы (DB2, Oracle, MS SQL server, Interbase и другие). В них представлениям пользователя соответствуют поименованные запросы (View) и хранимые процедуры (Stored Procedure), построенные на основе общей схемы данных. Физическое представление данных, как правило,

тесно связано с СУБД, поэтому перенос данных с одной платформы на другую возможен в виде экспорта-импорта.



В проектировании информационного обеспечения можно выделить:

1. Инфологическое проектирование – решение вопросов, связанных со смысловым содержанием данных:
 - о каких объектах и явлениях следует накапливать информацию;
 - какие их характеристики и взаимосвязи будут учитываться.
2. Даталогическое проектирование – решение вопросов связанных с представлением данных:
 - типы и форматы данных;
 - методы преобразования и смысловой интерпретации данных.

7.4.3. Инфологическое проектирование. Модель «Сущность-связь»

Инструментом инфологического проектирования является представление предметной области в виде множества сущностей и связей между ними. Это представление отражается в виде диаграмм "сущность-связь" (Entity-Relationship Diagram – ERD). Модель "Сущность-связь" [13] позволяет получить общее описание данных, не связанное с представлением данных в информационной системе.

Сущность – обобщенное представление о множестве подобных объектов реального мира. Все объекты, соответствующие некоторой сущности, описываются одинаковым набором атрибутов. Например, все поставщики имеют наименование, адрес, счет в банке и т.д. В базе данных должно быть предусмотрено хранение всех атрибутов для каждого экземпляра сущности. На диаграмме сущность изображается прямоугольником. Сущности

обычно именуются существительными. Их можно классифицировать следующим образом:

- роли (люди, организации),
- реальные предметы,
- события (договора, поставки),
- расположение.

Например, для регистрации торговых операций можно выделить следующие сущности: «Поставщик» (роль), «Заказ» (событие), «Товар» (реальные предметы), «Заказчик» (роль). Деление на сущности достаточно произвольно. Например, можно объединить поставщиков и заказчиков в одну сущность – организация. Результатом такого объединения будет единый инструмент описания и поставщиков и заказчиков.

На основе общности атрибутов и поведения сущностей может быть введена классификация сущностей. Для классификации сущностей вводят сущность – супертип, которая объединяет в одну сущность реальные объекты, соответствующие нескольким сущностям – подтипам (см. рис. 19). Например, роль супертипа может играть сущность "Организация" по отношению к подтипам "Поставщик" и "Заказчик". Соединение супертипа с подтипами на диаграмме выполняется линиями с выделением супертипа черточкой.

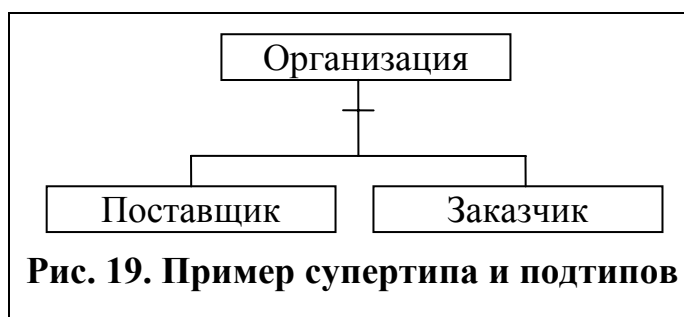


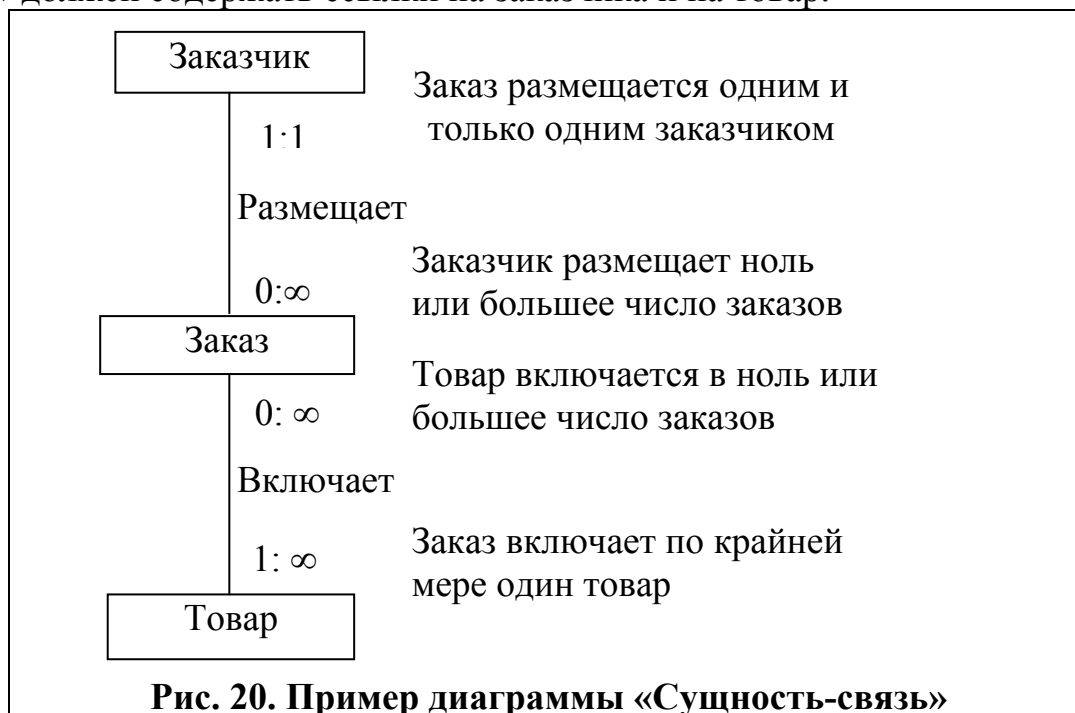
Рис. 19. Пример супертипа и подтипов

Связь – некоторая характеристика взаимодействия или ассоциации сущностей. Связь обычно именуется глаголом. Так же как сущность связь представляет множество зависимостей одного типа. Связи могут отражать постоянные зависимости сущностей (банк – клиент) или события, происходящие во времени (поступление денег на счет клиента в банке, факт поставки некоторого товара от поставщика заказчику). На диаграмме связь изображается линией, которая соединяет сущности (см. рис. 20).

Чаще всего встречаются бинарные связи. Участие каждой сущности в связи характеризуется мощностью – двумя числами: ординальное число указывает минимальное количество сущностей в связи, кардинальное – максимальное. Определение связей является основой для ссылочной целостности модели. Например, включение в модель связи между товаром и заказом означает, что в заказе невозможно указать товар, который не включен в экземпляры сущности «Товар».

Обычно связи не представляются данными, за исключением связи многие ко многим. Такую связь обычно преобразуют в сущность. Например, связь между «Заказчиком» и «Товаром» представлена на рис. 20 сущно-

стью «Заказ». Выделяют независимые сущности, которые не содержат ссылок на другие сущности. Следовательно, их описание не зависит от содержания остальной части данных. На рис. 20 такими сущностями являются «Заказчик» и «Товар». Зависимые сущности (такие как «Заказ») ссылаются на другие сущности (и в этом смысле зависят от них). Например, «Заказ» должен содержать ссылки на заказчика и на товар.



Среди множества атрибутов, описывающих некоторую сущность, можно выделить набор атрибутов, значения которых однозначно определяют один экземпляр сущности. Такой набор называют ключом. Среди множества ключей выделяют один, который систематически используют для ссылки на конкретный экземпляр сущности и называют первичным ключом. Набор атрибутов сущности, используемых для ссылки на экземпляр другой сущности, называют внешним ключом. Например, среди атрибутов сущности «заказ» должны быть внешние ключи – атрибуты первичных ключей «Товара» и «Заказчика».

В связи можно выделить родительскую сущность, на которую ссылается другая – зависимая от нее «дочерняя» сущность. На рис. 20 «Заказчик» является родительской, а «Заказ» дочерней сущностью.

В модели возможно появление рекурсивных связей – когда появляется цепочка связей сущности с собой. Например, сущность «Работник» может иметь атрибут «Непосредственный начальник», который будет ссылкой на другой экземпляр той же самой сущности.

Для построения модели "Сущность-связь" рекомендуются следующие действия:

- 1) Идентификация сущностей. Для идентификации полезными оказываются следующие приемы:
 - При интервьюировании пользователей выделяйте ключевые слова.
 - Просите пользователей определить вещи, о которых необходимо собирать, хранить и извлекать информацию.
 - Изучайте формы документов.
 - Исследуйте уже созданные и применяемые ИС.

Выясните количество экземпляров сущностей (десятки, сотни, ...). Придумайте простые звучные содержательные имена, имеющие экономический смысл. Старайтесь избегать аббревиатур. Имена сущностей должны быть уникальны.

- 2) Описание атрибутов. Для этого полезно изучать формы документов. При выборе имен атрибутов часто применяются уточнения (например, не просто ДАТА, а ДАТА ПОСТАВКИ).
- 3) Описание каждой сущности набором атрибутов.
- 4) Выделение возможных ключей и выбор первичных ключей. Ключи обязательно должны быть, если предполагается ссылаться на экземпляры сущностей. Однако, и в случае отсутствия ссылок ключи позволяют автоматически поддерживать уникальность определенного набора атрибутов. В качестве первичных ключей удобно выбирать поля, не несущие смысловой нагрузки, например, порядковый номер экземпляра сущности в базе. Тогда не придется изменять ссылки при изменении описания экземпляра сущности.
- 5) Идентификация связей. Выделяются связи, необходимые для обеспечения ссылочной целостности. При этом необходимо учитывать технологический аспект: в какой последовательности будет производиться регистрация данных.
- 6) Формулировка и запись бизнес-правил, сопровождающих добавление, корректировку, удаление данных.

Модель данных используется на всех этапах существования ИС. На этапе планирования выделяют информационные потребности всей организации в целом (информационно-ориентированный подход). Это позволяет выделить сущности, информация о которых играет значимую роль в деятельности организации (поставщики, заказы, товары,...).

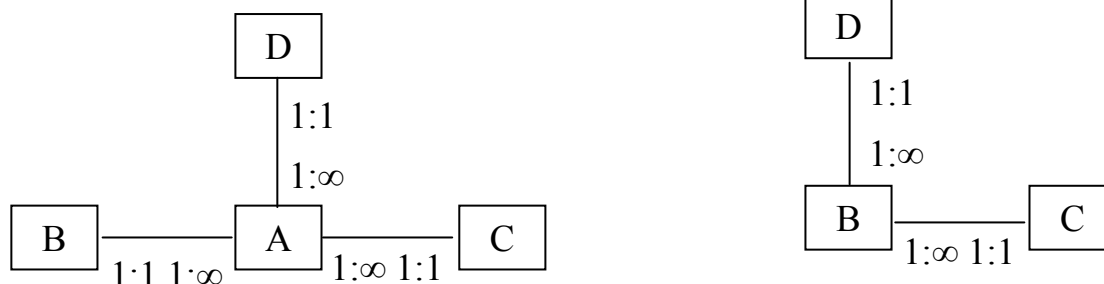
При проектировании подсистем выделяются соответствующие модели данных подсистем. На этапе конструирования разрабатывается модель "сущность-связь" подсистемы.

Важной частью работы по формированию модели данных всей информационной системы, объединяющей несколько подсистем, является согласование подмоделей данных (подсхем) с моделью данных организации в целом (см. рис. 21). Для этого используются следующие операции:

- 1) Установление идентичности. Два элемента идентичны, если они имеют одинаковое смысловое значение.
- 2) Агрегация – объединение связанных элементов модели в один сложный (адрес, банковские реквизиты).
- 3) Обобщение – разбиение множества сущностей на классы эквивалентности и введение сущностей соответствующих понятию класса. Создание таким образом иерархической структуры типов сущностей.

Проблема заключается в использовании разными пользователями разных терминов и описаний для одних и тех же понятий.

Представление первого пользователя Представление второго пользователя



Объединенное представление

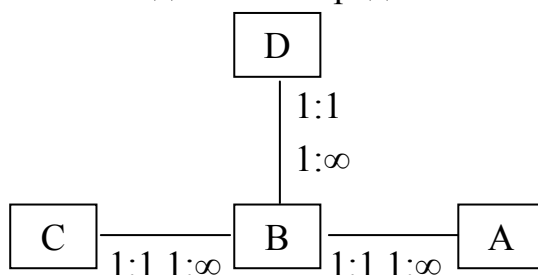


Рис. 21. Объединение представлений пользователей

7.4.4. Анализ событий (Event Analysis)

Анализ событий заключается в определении экономических событий, вызывающих добавление, чтение, изменение и удаление данных.

1-й шаг. Идентификация событий для сущностей. Результатом идентификации является таблица «Сущности-события», состоящая из следующих колонок:

- сущность,
- краткое описание события,
- имя события,
- тип действия: добавление, чтение, изменение или удаление данных,
- условия выполнения действия.

Одно событие может влиять на много сущностей, и одна сущность может быть связана со многими событиями. Идентификация событий требует привлечения пользователей.

Условия можно разбить на два класса:

- вытекающие из экономической практики (известны пользователям),
- обусловленные связями структур данных (определяются системными аналитиками): например, до добавления дочерней записи должна существовать родительская. Удаление родительской записи возможно, если с ней не связана ни одна дочерняя запись.

Для событий, вызывающих изменения, должны указываться атрибуты, которые могут изменяться.

2-й шаг. Консолидация общих событий. Проверяется таблица «Сущности – события» для выделения одинаково названных разных сущностей и по разному названных одинаковых событий. Строки таблицы группируются по сущностям для проверки жизненного цикла сущности – операции добавления, изменения, чтения и удаления должны выполняться в определенном порядке. Группировка таблицы по событиям позволяет для каждого события выделить действия над сущностями, определить функции и полномочия пользователей.

По результатам построения модели данных и анализа событий пересматривается описание потоков данных. При этом возможно:

- добавление входных данных для проверки условий (например, на существование родительской записи);
- добавление атрибутов;
- добавление новых операций (например, различных проверок).

7.4.5. Даталогическое проектирование

Содержанием даталогического проектирования является определение модели данных. Модель данных – это набор соглашений по способам представления сущностей, связей, агрегатов, систем классификации. Кроме этого, каждая модель данных определяет правила выполнения основных операций над данными: добавления, удаления, модификации, выборки.

Особое внимание при построении модели уделяют целостности и отсутствию избыточности данных. *Избыточность* – это многократное повторение одних и тех же данных. Например, рассмотрим ИС, в которой регистрируются торговые операции (см. рис. 20). В ИС регистрируются все факты продаж. Если в БД имеется несколько описаний одного и того же заказчика, то все экземпляры этих описаний, кроме одного будут избыточными. Однако, при регистрации заказа необходимо указывать заказчика. В результате при регистрации нескольких заказов одного заказчика в БД бу-

дет несколько ссылок на данного заказчика. Очевидно, что эти ссылки не являются избыточными. Кроме этого, несколько описаний одного и того же поставщика могут отражать изменение его реквизитов (адреса, расчетного счета в банке) с течением времени. Приведенные примеры показывают, что проблема избыточности не является такой тривиальной, как это кажется на первый взгляд, и требует специальных методов исследования.

Целостность – это непротиворечивость одних данных другим. Целостность зависит от степени избыточности данных. Например, если в БД имеется несколько описаний одного заказчика, то при изменении адреса в одном описании нарушается целостность данных. Однако, даже при избыточности данных может возникнуть нарушение целостности. Пусть из БД удаляются сведения о некотором заказчике. Теперь ссылки на этого заказчика в зарегистрированных до удаления продаж будут неправильными и также квалифицируются как нарушение целостности. Кроме целостности и избыточности при проектировании БД учитывают возможность быстрого выполнения запросов к БД и оптимального использования ресурсов, а также разграничение доступа для разных групп пользователей.

Для построения БД могут использоваться три модели: иерархическая, сетевая, реляционная [8]. Все эти модели отличаются в основном способами представления связей сущностей.

7.4.6. Иерархическая модель

В иерархической модели данные представлены в виде множества деревьев (прадеревьев). Дерево состоит из вершины, связанной произвольным количеством деревьев (поддеревьев). Вершины поддеревьев некоторой вершины называются ее потомками, а саму эту вершину – предком по отношению к ним. Вершина дерева, которое не является поддеревом ни для одной другой вершины, называется корнем дерева. Вершины, не имеющие поддеревьев, называются терминальными или листьями.

Для описания связи в иерархической модели необходимо включить в дерево сущности все ее поддеревья. Для примера с рис. 20 в описание заказчика кроме его атрибутов должны быть вставлены поддеревья, описывающие его заказы (см. рис. 22), которые, в свою очередь, должны содержать поддеревья, описывающие товар. При помощи деревьев можно описывать любые иерархические структуры. Проблемы появляются, если в модели данных появляются зависимости «многие ко многим, как между заказчиками и товарами. Действительно, в каждое дерево заказов необходимо включить поддерево описания товара. При этом описание товара будет дублироваться по числу его продаж.

Для уменьшения избыточности, связанной с представлением связей многие ко многим вводят данные типа указатель для создания виртуального дерева. Это решает проблему избыточности данных.



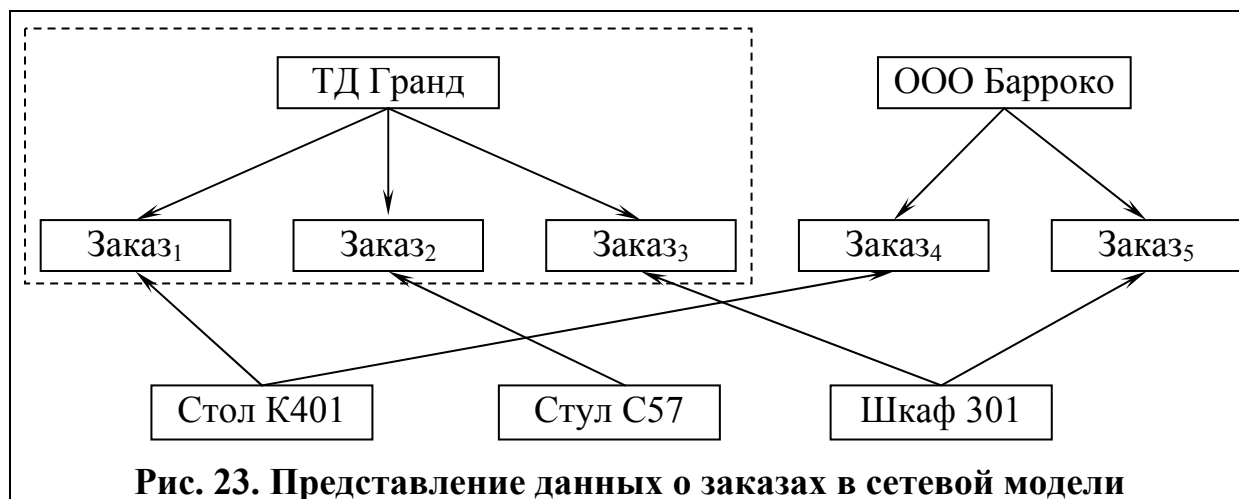
Поиск и обработка данных в иерархической модели выполняются в виде процедур просмотра деревьев, а в дереве в виде процедур обхода дерева. Эти процедуры зависят от положения сущности в иерархии сущностей. В примере на верхнем уровне расположены заказчики. Для обработки данных по заказчику достаточно найти и обработать описывающее его дерево. Для обработки данных по товару (расположенному на нижнем уровне иерархии) придется просматривать все деревья. Таким образом, даже виртуальные деревья не решают проблем, связанных с построением обработки и извлечения данных в иерархической модели.

7.4.7. Сетевая модель

Основным элементом сетевой модели является набор, который описывает двухуровневую структуру, состоящую из одного владельца и нескольких связанных с ним компонентов (на рис. 23 пример набора выделен пунктиром). Компоненты могут быть связаны одновременно с владельцами разных типов. На рис. 23 заказы имеют в качестве владельцев одного продавца и один товар. Эта конструкция позволяет реализовать связи «многие ко многим» без присущей иерархической модели избыточности.

Компонент некоторого набора может сам быть владельцем других компонентов и это позволяет описывать иерархические структуры. Например, владелец «предприятие» связан с компонентами – «цехами», которые являются владельцами компонент типа «бригада», а «бригада» владеет компонентами – «рабочими».

Операции поиска и обработки определяются в терминах просмотра наборов. Поиск должен начинаться с владельцев, расположенных на самом верхнем уровне иерархии, и продолжается среди компонент. Жесткая стратегия поиска является основным недостатком сетевой модели.



7.4.8. Реляционная модель

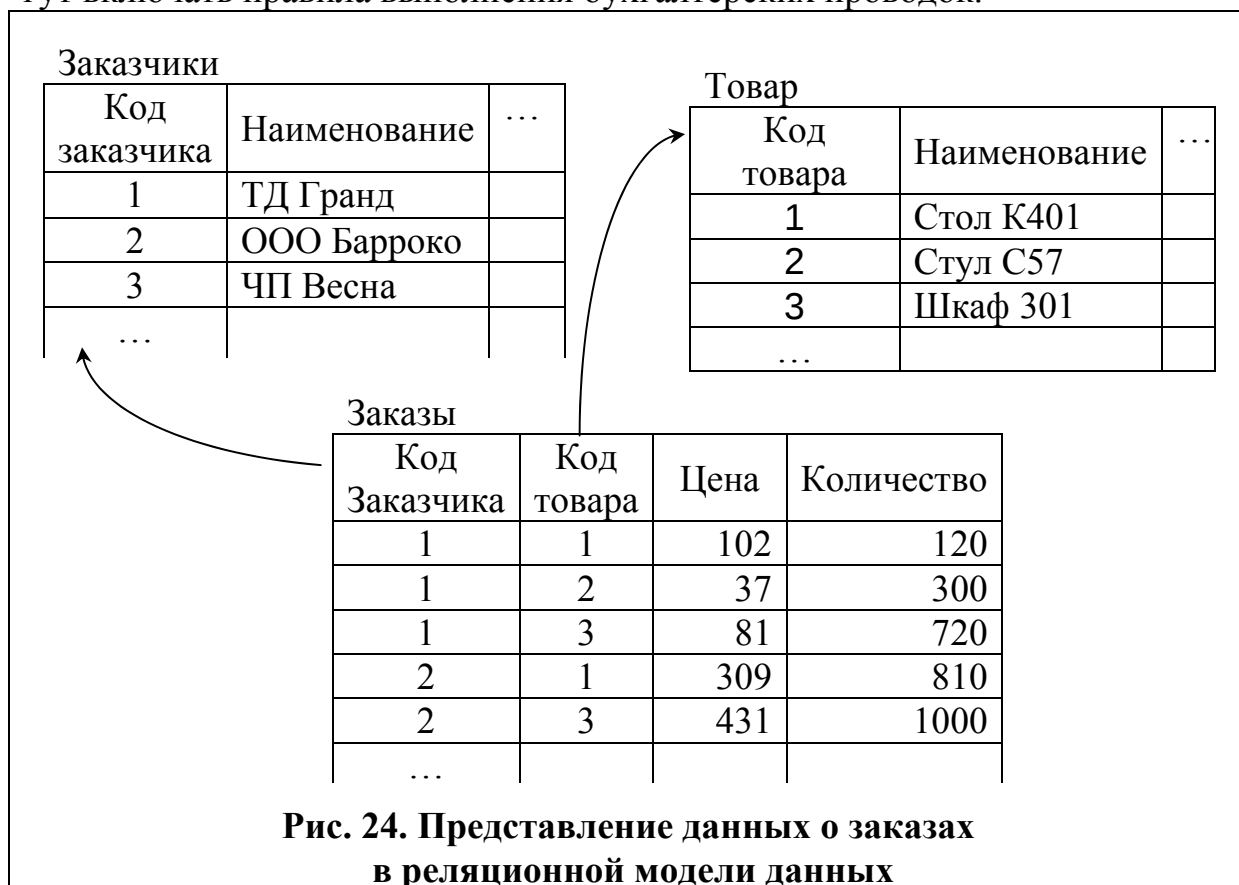
Основы реляционной модели данных были сформулированы Э.Ф.Коддом в 1970 г. Основной структурой этой модели является отношение – таблица со следующими свойствами:

1. Каждый столбец содержит информацию одного типа.
2. Ячейки – поля таблицы не содержат агрегатов (структур или массивов) данных.
3. Таблицы не содержат одинаковых строк.
4. Порядок строк и столбцов не имеет значения. Все операции используют содержательную сторону данных, а не их расположение внутри таблицы.

Для описания связей вводятся первичные ключи, позволяющие указывать ровно одну строку (кортеж) таблицы. Значение ключа может использоваться для ссылки в другой таблице – это использование и определяет связь двух таблиц. Поскольку первичный ключ играет ведущую роль в описании связей и поиске данных, размер ключа стараются сделать минимальным для оптимизации поиска. Это приводит к использованию номеров или кодов в качестве первичных ключей. На рис. 24 приведено представление данных о заказах в реляционной модели. Стрелками показаны ссылочные связи таблиц.

Для правильного связывания не должно существовать неопределенных ссылок – каждая ссылка должна указывать на определенную запись. Почти все реляционные СУБД поддерживают ссылочную целостность автоматически. Кроме ссылочной целостности должны быть определены и описаны другие ограничения и зависимости. Реляционные СУБД позволяют сделать это в виде хранимых процедур и триггеров (последовательностей команд запускаемых автоматически при изменении содержимого таблицы). Таким

образом, реляционная модель должна включать таблицы, связи – ограничения ссылочной целостности, прочие ограничения на значения записей, а также ряд общезначимых функций, реализованных в виде процедур и триггеров и называемых бизнес-правилами. Например, бизнес-правила могут включать правила выполнения бухгалтерских проводок.



Для выбора данных в реляционной модели используют операции соединения таблиц (обычно к записи дочерней таблицы добавляются поля записи родительской таблицы, на которую ссылается запись дочерней таблицы, хотя могут употребляться и более сложные варианты соединений), проекции (выбор лишь некоторых полей для таблицы-результата), фильтрации (выбор записей, удовлетворяющих условию) и некоторых других, образующих вместе реляционную алгебру. Наибольшее распространение получил язык SQL, который поддерживает большинство современных СУБД. Кроме перечисленных возможностей реляционной алгебры в него встроены операции группировки и агрегирования (вычисления по каждой группе сумм, минимумов, максимумов, средних значений и других подобных функций), а также возможности применения подзапросов.

Реляционный подход часто критикуют за отсутствие иерархии абстракций данных и объектно-ориентированных средств.

7.5. Определение набора приложений. Проектирование интерфейса пользователя, структуры и логики работы программы

Выделение приложений выполняется на основе анализа событий, прав и обязанностей персонала. Для каждого приложения указывают назначение, персонал, использующий приложение, функции, автоматизируемые приложением, и регламент их выполнения, взаимодействие с серверами.

Для описания функций можно использовать стандартный документ «Описание алгоритма (проектной процедуры)» (см. Приложение). Существует достаточно много средств описания алгоритмов.

- блок-схемы,
- HIPO (Hierarchy+Input-Process-Output) диаграммы,
- деревья решений,
- диаграммы Nassi-Shneiderman,
- таблицы решений,
- структурный псевдокод,
- конечные автоматы (граф смены состояний).

Какой бы вариант не был избран, важно, чтобы описание было точным и полным.

Проектирование интерфейса в современных инструментальных системах выполняется специальным графическим редактором, который наряду с изображением генерирует программу, создающую интерфейс. Поэтому достаточно ограничиться общепринятыми рекомендациями.

Рекомендации по проектированию входных и выходных форм

1. Минимизируйте объем ввода:
 - Вводите только переменные данные. Не вводите постоянные.
 - Не вводите значения, которые вычисляются.
 - Используйте подходящие коды (например, «м» и «ж» вместо «мужской» и «женский»).
2. Формы должны быть удобны для заполнения:
 - Размещайте подсказки.
 - Минимизируйте ручные действия (используйте переключатели, радиокнопки, поля с выпадающими списками).
3. Технология ввода должна быть простой: Последовательность размещения полей в форме должна совпадать с последовательностью ввода.
4. Выходные формы должны быть просты для понимания и использования:
 - Должен быть заголовок документа.
 - Должны быть заголовки колонок.
 - Должны быть пояснения к сокращениям и условным обозначениям.
 - Не следует использовать жаргон.
5. Включайте в форму дату и время формирования.

6. При проектировании выходной формы необходимо учитывать ее использование.

Рекомендации по проектированию контрольных функций

1. Регистрируйте объемы введенных данных в журналах.
2. Контролируйте вводимые данные:
 - полноту (все ли поля заполнены?),
 - допустимые значения полей,
 - допустимые комбинации значений полей,
 - используйте шаблоны для уменьшения ошибок ввода,
 - применяйте контрольные суммы.
3. Совмещайте изготовление первичного документа и ввода данных.
4. Контролируйте время и объем операций вывода.
5. Контролируйте распределение выходных документов по пользователям.
6. Используйте контрольные суммы в выходных документах.

Рекомендации по проектированию интерфейса

1. Грамотно применяйте компоненты интерфейса:
 - пассивный и активный диалоги,
 - главные, дочерние и диалоговые окна,
 - командные кнопки,
 - меню,
 - блоки радиокнопок.
 - переключатели,
 - поля ввода,
 - поля со списками выбора.
2. Система должна подсказывать, что пользователь должен сделать дальше.
3. Поля ввода должны размещаться в ожидаемых пользователем зонах и единообразно.
4. Диалог должен быть ограничен окном.
5. Визуальные эффекты должны быть строго дозированы.
6. Умолчания должны быть размещены в полях.
7. Выводите предупреждения в случае опасных действий.
8. Предоставляйте возможность отмены изменений.
9. Используйте в подсказках простые правильные предложения.
10. Избегайте сокращений.
11. Последовательно используйте термины (например, используйте или термин "редактирование", или "изменение").
12. Добивайтесь точности инструкций.

Часто не хватает времени на то, чтобы выполнить работу правильно. Однако, время на переделку всегда находится.

8. Реализация и сопровождение

На стадии реализации создаются структуры данных, программы и документация к ним, инструкции по использованию программ, выполняется обучение пользователей и запускаются в эксплуатацию готовые программы. Реализация включает:

- создание и тестирование сети и базы данных;
- создание, тестирование и документирование отдельных программ;
- сбор, установка, тестирование и документирование системы в целом;
- ввод в эксплуатацию (разработка инструкций пользователям, обучение, консультирование).

Вследствие быстрого развития инструментальных средств разработки информационных систем сложно разделить этапы конструирования и реализации. Практически создание базы данных и приложений (в виде прототипов) уже выполняется на этапе конструирования по результатам разработки соответствующих описаний.

Стандартные документы стадии реализации приведены в приложении. Некоторые из них сохранили актуальность: «Описание системы (подсистемы)», «Описание организационной структуры», «Описание алгоритма (проектной процедуры)», «Руководство пользователя», организационно-распорядительные документы; другие – (документация по техническому обеспечению) редко разрабатываются в рамках подсистемы, так как описывают свойства системы в целом (например, локальная сеть); третьи – (документы по информационному обеспечению, «Описание программы») достаточно сложно адаптировать, вследствие глубоких изменений технологий программирования.

Какой бы стандарт не был выбран разработчиками, он должен содержать:

- конструктивные описания:
 - описание архитектуры системы (подсистемы): приложения, серверы, регламенты решения задач;
 - описание информационного обеспечения: структуры данных, ограничения, бизнес-правила;
 - описание математического и программного обеспечения: математические модели, структуры программ (например, объектные и/или компонентные);
- эксплуатационные документы:

- общий регламент управления в условиях автоматизации;
- инструкции пользователям (порядок и правила использования информационных технологий);
- инструкции администраторам (управление полномочиями, развертывание информационного и программного обеспечения, поддержка работоспособности, безопасности и надежности системы, восстановление системы в случае аварий и сбоев);
- руководства программистам по сопровождению и развитию.

Ввод системы в опытную (промышленную) эксплуатацию обязательно должен быть оформлен юридически соответствующими документами (требования к актам о приемке в опытную (промышленную) эксплуатацию и к протоколам испытаний и согласований приведены в приложении). В соответствующие комиссии должны входить представители заказчика, исполнителя и ключевые пользователи.

Для обеспечения достоверности испытаний заказчиком должен быть составлен контрольный пример: варианты исходных данных и значения показателей, которые должны быть получены программным обеспечением по этим данным. Однако, заказчик зачастую пренебрегает этим правилом и испытание производится по контрольному примеру исполнителя.

В случае положительных результатов испытаний издается приказ о начале опытной эксплуатации, в котором должны быть определены сроки, пользователи и регламент использования информационной системы, а также состав и порядок работы соответствующей комиссии.

В период опытной эксплуатации выявляются и оформляются протоколом согласования все ошибки проектирования, в том числе и ошибки заказчика, согласованные ошибки исправляются. Возможно также дополнение системы новыми функциональными возможностями. В соответствии с изменениями вносятся изменения в конструктивные и эксплуатационные документы.

Опытная эксплуатация заканчивается сдачей в промышленную эксплуатацию, которая оформляется соответствующими актом и приказом.

Сопровождение заключается в поддержании в рабочем состоянии программного и информационного обеспечений. На этой стадии выполняются консультации и дополнительное обучение пользователей, восстановление после сбоев, исправление ошибок и адаптация программы к изменениям в условиях эксплуатации.

Исправление ошибок предполагает:

- 1) определение проблемы,
- 2) понимание программ,
- 3) изменение и тестирование в сравнении с предыдущей версией,
- 4) изменение документации.

Адаптация программы обычно включает:

- 1) анализ требований,
- 2) написание новых простых программ,
- 3) реструктуризация файлов и баз данных,
- 4) анализ и переделка комплекса программ.

Чтобы выносить ребенка, нужно девять месяцев, независимо от того, сколько женщин будет к этому привлечено.

9. Организация процесса разработки информационной системы

Организация разработки и внедрения информационной системы включает следующее:

- Управление основными характеристиками проекта: цели, назначение, автоматизируемые функции, свойства программного, информационного и других видов обеспечения.
- Организация коллективной работы над проектом.
- Управление процессом проектирования: выбор модели жизненного цикла (определение стадий и этапов разработки), оценка ресурсов (трудовых, временных, стоимостных), необходимых для выполнения работ на каждом этапе, оценка и управление рисками, оценка целесообразности продолжения работ в запланированном объеме, составление календарных графиков работ и отслеживание их выполнения.

9.1. Управление основными характеристиками проекта

Для успеха проекта принципиально важно соблюдать приоритет требований системы управления. Для этого необходимо добиться с одной стороны понимания задач управления проектировщиками, с другой – понимания менеджерами стоимости информационных технологий и их значимости для эффективного решения задач управления. Привлечение проектировщиков к реинжинирингу бизнес-процессов, а менеджеров к принятию решений по информационным технологиям является ключом к созданию действительно полезной и нужной информационной системы.

Довольно часто пользователи под впечатлением «легкости» применения компьютерных программ не осознают сложности создания информационной системы, которая охватывает множество подразделений и объединяет усилия многих пользователей, имеющих зачастую противоречивые интересы. С другой стороны, менеджеры плохо представляют новые возможности, которые предоставляют информационные технологии, и поэто-

му склонны ограничиться автоматизацией тех функций, с которыми они справлялись раньше «вручную».

Наиболее частой ошибкой проектировщиков является построение системы на основании приблизительных представлений о целях, задачах и технологии управления.

На качестве проектирования положительно сказывается правильное разбиение проекта на итерации. В первую очередь должны быть автоматизированы ключевые операции управления. К сожалению, наиболее часто приходится начинать с учетных задач, которые обеспечивают накопление данных, необходимых для решения задач управления. Необходимо, чтобы первые очереди системы действительно облегчали работу операционистов (ускоряли подготовку документов, автоматизировали подведение промежуточных итогов, уменьшали количество ошибок) и, кроме этого, позволяли извлекать информацию полезную для управления.

9.2. Организация коллективной работы над проектом

Первоначально коллектив разработчиков строился по стандартной схеме управления предприятием. В основе этой схемы лежит принцип единоначалия с элементами авторитарной власти. Основными характеристиками такой организации являются низкая демократичность и низкая мобильность персонала, поэтому она плохо подходит для объединения в единый коллектив высокоинтеллектуальных исполнителей. Этой жесткой иерархической структуре нашли альтернативу в виде бригады с коллективной ответственностью за результат работы.

«Бригада главного программиста» включает руководителя работ, главного программиста, второго пилота, тестировщика, редактора, проблемных программистов. Руководитель работ осуществляет взаимодействие с пользователем, определит цели подцели, утверждает все проектные решения.

Проектирование ведется главным программистом и вторым пилотом. Такая система весьма эффективна, так как второй пилот постоянно подвергает критике решения главного программиста и испытывает прочность его концепций, а кроме того, предлагает свои.

Тестировщик оказался очень полезной фигурой и по важности сравнился с главным программистом. Именно тестировщик, и никто другой, лучше всех знает текущее состояние проекта, и с вопросами, насколько готов модуль, обращаются не к автору модуля, а к тестировщику. Критерием работы тестировщика является количество непройденных модулем тестов – чем больше, тем лучше.

Редактор разрабатывает проектную документацию (оформляет идеи главного программиста и второго пилота) и документацию пользователя.

Такой модели присуща довольно высокая степень мобильности, однако демократизация остается по-прежнему низкой (поскольку главный программист единолично принимает все принципиальные решения по проекту). Вследствие этого проект очень сильно зависит от компетенции и личных качеств главного программиста.

В последнее десятилетие появилась модель команды-сообщества (community team), ярким примером которой является команда Microsoft [3] и методология разработки программных продуктов Microsoft Solutions Framework [1]. Это наиболее демократичная модель, поскольку в ней нет явно выделенного центра. Команда-сообщество – это команда равных. В подобной команде взаимодействие участников проекта происходит на уровне сотрудничества. В группу MS Framework входят:

- **Менеджер продукта** – представляет заказчика и его интересы, вместе с менеджером программы находит компромисс между функциями проекта с одной стороны и сроками и бюджетом с другой. У заказчика представляет группу, объясняя сроки и бюджет работ, достигает с ним компромисс о финансировании работ.
- **Менеджер программы** – организует выполнение всех стадий и этапов разработки для выпуска нужного продукта в нужные сроки, координирует при этом деятельность других участников группы. Принимает и следит за выполнением графика работ (успешный способ составления графиков – «снизу-вверх» на основе графиков исполнителей) и всего бюджета проекта. Отвечает за набор функциональных возможностей.
- **Разработчики** выбирают применяемые технологии и создают продукт. Оценивают затраты на создание.
- **Тестировщик** выполняет испытание продукта для оценки работоспособности отдельных функций и продукта в целом. Разрабатывает стратегию, планы, графики и сценарии тестирования.
- **Инструктор** отвечает за выпуск удобного и полезного для пользователей продукта. Проектирует документацию пользователя, систему подсказок и, при необходимости, учебные курсы.
- **Логистик** отвечает за доставку, поддержку и сопровождение продукта, составляет график развертывания, порядок передачи продукта, готовит продукт к развертыванию.

Допускается совмещать роли с учетом отсутствия противоположных интересов: роль разработчика не совмещается ни с одной другой, противоположны интересы у менеджера продукта и менеджера программы. Роль тестировщика не совместима с ролью разработчика.

К преимуществам данной модели можно отнести: высокую функциональную гибкость, высокую масштабируемость и производительность команды. Однако, на пути воплощения в жизнь подобной модели встают не-

сколько существенных преград: подбор участников должен вестись не только исходя из компетенции, но и с учетом психологической и социокультурной совместимости, а также с учетом умения работать в команде (артельного духа); для формирования подобной команды нужны равные (одинаково квалифицированные и равно заинтересованные) участники.

9.3. Экономические показатели проектирования

Управление процессом проектирования начинается с выбора модели жизненного цикла, разбиения всего проекта на очереди (итерации), формирования коллектива разработчиков. Затем выполняется определение финансовых, временных и других ресурсов.

Можно выделить следующие виды расходов, связанных с информационной системой:

1. Расходы на разработку и внедрение:

- зарплата разработчиков;
- стоимость машинного времени;
- накладные расходы;
- материалы;
- приобретение вычислительной техники (ПК, серверы, сетевое оборудование);
- покупка программного обеспечения;
- обучение пользователей;
- стоимость разработки:

2. Расходы на эксплуатацию:

- трудовые затраты (зарплата администраторов, группы сопровождения, операторов);
- амортизация;
- материалы (картриджи, диски, бумага...);
- ведение информационной базы;
- накладные расходы (до 20%).

В определении ресурсов наиболее сложным является определение затрат труда (см. классические исследования в этой области [4, 20]). Общей практикой оценки трудозатрат является применение экспертных оценок исполнителей [1, 3, 12]) с выделением резервов от 30 до 50%. Эти оценки используются для построения календарных графиков и определения затрат на оплату труда.

Источниками доходов от внедрения и модернизации информационных систем и технологий могут быть следующие:

- качественное улучшение процесса управления,
- снижение трудовых затрат (сокращение штатов),

- увеличение производительности труда,
- снижение стоимости инфраструктуры,
- уменьшение неопределенности и рисков [14],
- уменьшение совокупной стоимости владения (Total Cost of Ownership – TCO).

Для определения инвестиционной привлекательности проекта используют традиционные оценки. Для оценки эффективности проекта анализируются инвестиции (расходы) I и доходы $CF(i)$. При этом используют следующие показатели:

Чистая приведенная величина дохода (NPV – Net Present Value) – оценка дохода за некоторый период времени T , пересчитанная на момент вложения по процентной ставке γ

$$NPV = -I + \sum_{i=1}^T CF(i) / (1 + \gamma)^i$$

Срок окупаемости проекта (BRP – Payback Period) – это время, за которое приведенная величина инвестиций сравняется с приведенной суммой доходов, находится как решение уравнения

$$NPV = -I + \sum_{i=1}^T CF(i) / (1 + \gamma)^i = 0$$

относительно T .

Индекс прибыльности (PI – Profitability Index)

$$PI = \frac{\sum_{i=1}^T CF(i) / (1 + \gamma)^i}{I}$$

Внутренняя норма прибыльности (IRR – Internal Rate of Return) – это такая процентная ставка, при которой чистая приведенная величина дохода за все будущие периоды равна 0, т.е. когда приведенные доходы за все будущие периоды будут равны инвестициям. Находится как решение уравнения

$$-I + \sum_{i=1}^{\infty} CF(i) / (1 + \gamma)^i = 0$$

относительно γ .

Кроме оценки инвестиционной привлекательности используют другие методологии оценки информационных технологий см. обзор Т.Мейлора [19].

ПРИЛОЖЕНИЕ

Требования к содержанию проектных документов

Пояснительная записка к эскизному (техническому) проекту

- 1) Общие положения:
 - наименование;
 - разработчики;
 - цели, назначение и области использования системы;
 - соответствие нормам техники безопасности и нормативным документам;
 - сведения о НИР, изобретениях, открытиях, использованных при разработке;
 - очередность создания системы.
- 2) Описание процесса деятельности (состав процедур, требования к организации работ).
- 3) Основные технические решения:
 - структура системы, связь между структурными единицами;
 - связь со смежными системами;
 - функционирование и диагностирование;
 - состав, квалификация, функции, режим работы персонала системы;
 - состав функций, комплексов задач (задач), реализуемых системой (подсистемой);
 - комплекс технических средств;
 - состав, объем, способы организации информации, виды носителей информации, входные и выходные документы и сообщения, последовательность обработки;
 - состав программных средств, языки, алгоритмы и методы их реализации.
- 4) Мероприятия по подготовке объекта автоматизации к вводу системы в действие:
 - по приведению информации к виду, пригодному для обработки на ЭВМ;
 - по обучению и проверке квалификации персонала;
 - по созданию необходимых подразделений и рабочих мест;
 - по изменению объекта автоматизации.

Общее описание системы

- 1) Назначение системы:
 - вид автоматизируемой деятельности;
 - перечень объектов автоматизации;

- перечень функций, реализованных системой;
- 2) Описание системы:
 - структура системы и назначение ее частей;
 - сведения о системе в целом и ее частях, необходимых для обеспечения эксплуатации системы;
 - описание функционирования системы и ее частей.
- 3) Связи с другими системами:
 - перечень систем;
 - описание связей;
 - регламент связей;
 - связи с подразделениями объекта автоматизации.
- 4) Описание подсистем (при необходимости и подобно описанию системы).

Описание организационной структуры

- 1) Изменения в организационной структуре системы управления: проектные решения по изменению и их обоснования; изменение во взаимосвязях.
- 2) Организация подразделений: описание организационной структуры и функций создаваемых подразделений; описание регламента работ; категории работников и их количество.
- 3) Реорганизация подразделений.

Схема организационной структуры

- 1) Состав подразделений, обеспечивающих функционирование системы или использующих информацию из системы.
- 2) Основные функции и связи, а также их подчиненность.

Схема функциональной структуры

- 1) Элементы функциональной структуры (функции и/или задачи (комплексы задач)); действия, выполняемые автоматически и вручную.
- 2) Информационные связи между элементами и с внешней средой с указанием содержания сообщений, связи других типов (вложенности, подчинения и др.).
- 3) Детализация (при необходимости).

Описание автоматизируемых функций

- 1) Исходные данные:
 - перечень исходных материалов и документов для разработки функциональной части системы;

- особенности объекта автоматизации, влияющие на проектные решения по автоматизируемой функции;
 - сведения о системах, связанных с разрабатываемой системой, и об информации, которой они обмениваются;
 - информационная модель объекта.
- 2) Цели системы и автоматизированные функции (описание функций).
 - 3) Характеристики функциональной структуры:
 - перечень подсистем системы с указанием функций (задач);
 - описание процесса выполнения функции (при необходимости);
 - пояснения к разделению функции на действия, выполняемые техническими средствами и человеком.
 - 4) Типовые решения (при наличии).

Описание постановки задачи (комплекса задач)

- 1) Характеристики комплекса задач:
 - назначение;
 - перечень объектов, при управлении которыми решается данный комплекс;
 - периодичность и продолжительность решения;
 - условия прекращения использования (при необходимости);
 - связи с другими комплексами;
 - распределение действий между персоналом и техническими средствами.
- 2) Выходная информация:
 - перечень и описание выходных сообщений;
 - идентификатор;
 - форма (документ, видеокадр, файл, ...);
 - периодичность выдачи;
 - получатели и назначение;
 - перечень и описание структурных единиц сообщений, имеющих самостоятельное значение.
- 3) Входная информация:
 - перечень и описание входных сообщений;
 - наименование;
 - точность (при необходимости);
 - источник (документ, устройство, кодограмма, файл, ...);
 - идентификатор.

Описание технологического процесса обработки данных

- 1) Технологический процесс сбора и обработки данных на периферийных устройствах при децентрализованной обработке данных:

- состав и последовательность выполнения операций по сбору, регистрации, подготовке, контролю, передаче, обработке и отображению информации;
 - перечень документации на каждую операцию.
- 2) Технологический процесс обработки данных на ВЦ:
- состав и последовательность выполнения операций по приему, контролю, обработке, хранению, выдаче данных и других;
 - перечень документации на каждую операцию.

Схема автоматизации

- 1) Упрощенное изображение объекта или его части.
- 2) Средства технического обеспечения, участвующие в процессе, отображенном на схеме, за исключением вспомогательных.
- 3) Функциональные связи между средствами технического обеспечения.
- 4) Внешние функциональные связи с другими средствами технического обеспечения.
- 5) Таблица условных обозначений, не предусмотренных стандартами.

Описание комплекса технических средств

- 1) Общие положения (исходные данные для проектирования).
- 2) Структура комплекса технических средств:
 - обоснование выбора структуры комплекса технических средств;
 - описание функционирования комплекса технических средств, в том числе в пусковых и аварийных режимах;
 - описание размещения технических средств;
 - обоснование применения и требования к изготавливаемым в индивидуальном порядке технических средств;
 - обоснование методов защиты технических средств, в том числе от несанкционированного доступа;
 - оценка надежности технических средств.
- 3) Средства вычислительной техники:
 - обоснование и описание решений по выбору ЭВМ,
 - периферийных устройств;
 - структурная схема технических средств на ВЦ и рабочих местах;
 - расчет числа технических средств и потребностей в машинных носителях;
 - обоснование численности персонала, обслуживающего технические средства;
 - оснащение рабочих мест персонала, включая описание рабочих мест и расчет площадей;

- особенности функционирования технических средств в пусковом, нормальном и аварийном режимах.

4) Аппаратура передачи данных:

- выбор каналов связи и расчет их числа, технических средств сопряжения с каналами связи;
- размещение абонентов и объемно-временные характеристики передачи данных;
- надежность, достоверность и другие характеристики.

Описание информационного обеспечения

- 1) Состав информационного обеспечения (наименование и назначение всех баз и наборов данных).
- 2) Организация информационного обеспечения:
 - принципы организации информационного обеспечения;
 - обоснование выбора носителей и принципы распределения данных по типам носителей;
 - виды и методы контроля;
 - решения по обеспечению информационной совместимости с другими системами.
- 3) Организация сбора и передачи данных:
 - источники и носители информации с указанием интенсивностей и объемов;
 - общие требования к сбору, передаче, контролю и корректировке.
- 4) Построение системы классификации и кодирования.
- 5) Организация внутримашинной информационной базы:
 - принципы построения, характеристики состава и объема;
 - описание структуры на уровне баз данных, с описанием характера взаимосвязей баз данных и указанием функций системы, при реализации которых используют каждую базу данных, характеристики данных, содержащихся в каждой базе данных.
- 6) Организация немашинной информационной базы: состав, объем, принципы построения, связи с функциями системы.

Описание организации информационной базы

- 1) Описание внутримашинной информационной базы: логическая структура (состав, форматы, взаимосвязи); перечни БД и взаимосвязи между ними; физическая структура (расположение данных на носителях).
- 2) Описание немашинной информационной базы: логическая структура (состав, форматы, взаимосвязи); перечень документов и сообщений с указанием функций системы, при реализации которой используют до-

кумент или сообщение; физическая структура (расположение данных на носителях).

- 3) Организация ведения информационной базы: для внутримашинной базы – последовательность процедур с регламентом, средствами защиты от разрушения и несанкционированного доступа для немашинной базы – последовательность процедур по маршруту движения документов.

Описание массивов информации

- 1) Наименование.
- 2) Обозначение.
- 3) Носитель.
- 4) Перечень реквизитов в порядке следования в записях с указанием: обозначения, длины и диапазона изменения, логических и семантических связей.
- 5) Оценка объема.
- 6) Другие характеристики.

Описание алгоритма (проектной процедуры)

- 1) Назначение и характеристика:
 - назначение;
 - ссылка на документ с постановкой задачи, для решения которой он предназначен;
 - сведения об объекте и влияние решения на объект;
 - ограничения, условия применения, качество решения;
 - требования к данным.
- 2) Используемая информация (массивы).
- 3) Результаты решения (массивы).
- 4) Математическое описание.
- 5) Алгоритм решения:
 - последовательность этапов счета;
 - точность;
 - контроль достоверности;
 - связи между частями и операциями;
 - порядок расположения значений в выходных документах.

Алгоритм должен предусматривать все ситуации. Следует использовать обозначения таблиц, файлов, реквизитов, документов, и.т.д.

Акт завершения работы

- 1) Наименование завершенной работы.
- 2) Список представителей разработчика и заказчика, составивших акт.
- 3) Дата завершения работы.

- 4) Наименование документов, на основании которых производилась работа.
- 5) Основные результаты завершенной работы.
- 6) Заключение о результатах завершенной работы.

Акт о приемке в опытную эксплуатацию

- 1) Наименование АС или ее части, принимаемой в опытную эксплуатацию, и соответствующего объекта автоматизации.
- 2) Наименование документа, на основании которого разработана АС.
- 3) Состав приемочной комиссии и основания для ее работы.
- 4) Период времени работы комиссии.
- 5) Наименования разработчика и исполнителя.
- 6) Состав функций, принимаемых в опытную эксплуатацию.
- 7) Перечень составляющих технического, программного, информационного и организационного обеспечений, принимаемых в опытную эксплуатацию.
- 8) Перечень документов, предъявляемых комиссии.
- 9) Оценка соответствия принимаемой АС техническому заданию.
- 10) Основные результаты приемки в опытную эксплуатацию.
- 11) Решение комиссии о принятии АС в опытную эксплуатацию.

Акт о приемке в промышленную эксплуатацию

- 1) Наименование АС или ее части, принимаемой в промышленную эксплуатацию, и соответствующего объекта автоматизации.
- 2) Наименование документа, на основании которого разработана АС.
- 3) Состав приемочной комиссии и основания для ее работы.
- 4) Период времени работы комиссии.
- 5) Наименования разработчика и исполнителя.
- 6) Состав функций, принимаемых в промышленную эксплуатацию.
- 7) Перечень составляющих технического, программного, информационного и организационного обеспечений, принимаемых в промышленную эксплуатацию.
- 8) Перечень документов, предъявляемых комиссии.
- 9) Заключение о результатах опытной эксплуатации.
- 10) Оценка соответствия принимаемой АС техническому заданию.
- 11) Краткая характеристика и основные результаты выполненной работы по созданию АС.
- 12) Оценка научно-технического уровня АС
- 13) Оценка экономической эффективности АС.
- 14) Решение комиссии о принятии АС в эксплуатацию.
- 15) Рекомендации комиссии по дальнейшему развитию системы.

Протокол испытаний

- 1) Наименование объекта испытаний.
- 2) Список должностных лиц, проводивших испытание.
- 3) Цель испытания.
- 4) Сведения о продолжительности испытания.
- 5) Перечень пунктов ТЗ, на соответствие которым проводились испытания.
- 6) Сведения о результатах наблюдений.
- 7) Сведения об отказах, сбоях и аварийных ситуациях, возникших в процессе испытаний.

Протокол согласования

- 1) Перечень рассмотренных отклонений с указанием документа, устанавливающего требования.
- 2) Список должностных лиц, составивших протокол.
- 3) Обоснование принятых отклонений.
- 4) Перечень согласованных отклонений и срок внесения необходимых изменений в техническую документацию.

Список использованной литературы

1. Microsoft Corporation. Принципы проектирования и разработки программного обеспечения. Учеб. курс MCSD. М.: Изд.-торг. дом «Русская редакция», 2000.
2. Автоматизированные информационные технологии в экономике: Учеб./Под ред. Г.А.Титоренко. М.: Компьютер, ЮНИТИ, 1998.
3. Аджиев В. MS: корпоративная культура разработки ПО/ В.Аджиев// Открытые системы. 1998. № 1 (<http://www.osp.ru/os/1998/01/45.htm>).
4. Брукс Ф. Мифический человеко-месяц или как создаются программные системы/ Ф.Брукс. СПб.: Символ-Плюс, 1999.
5. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на С++/ Г.Буч. М.: ЗАО «Издательство Бином»; СПб.: «Невский диалект», 1999.
6. Буч Г. Язык UML. Руководство пользователя/ Г.Буч, А.Джекобсон, Д.Рамбо. М.: ДМК, 2000.
7. Глушков В.М. Введение в АСУ/ В.М.Глушков. Киев: Техника, 1974.
8. Дейт К.Дж. Введение в системы баз данных: 6-е изд./ К.Дж.Дейт. М.: Вильямс, 2000.
9. Информационная технология: Комплекс стандартов и руководящих элементов на автоматизированные системы. М., 1991.
10. Калянов Г.Н. CASE-технологии:Консалтинг в автоматизации бизнес-процессов/ Г.Н.Калянов. М.: Горячая линия – Телеком, 2000.
11. Липаев В.В. Документирование и управление конфигурацией программных средств. Методы и стандарты/ В.В.Липаев. М.: СИНТЕГ, 1998.
12. Лобанов С., Шамрай Д., Калишев К. Экстремальное программирование и управление проектами/ С.Лобанов, Д.Шамрай, К.Калишев// Открытые системы. 2002. № 9 (<http://www.osp.ru/os/2002/09/043.htm>).
13. Маклаков С.В. BPWin и ERWin. CASE-средства разработки информационных систем/ С.В.Маклаков. М.: Диалог-МИФИ, 1999.
14. Михайловский Н. Архитектура информационной системы, оценка рисков и совокупная стоимость владения/ Н.Михайловский// Директор ИС. 2002. № 6 (http://www.osp.ru/cio/2002/06/021_1.htm).
15. Мишенин А.И. Теория экономических информационных систем/ А.И.Мишенин. М.:Финансы и статистика, 2002.
16. Ойхман Е.Г. Реинжиниринг бизнеса: реинжиниринг организаций и информационные технологии/ Е.Г.Ойхман, Э.В.Попов. М.:Финансы и статистика,1997.
17. Смирнова Т.Н. Проектирование экономических информационных систем: Учеб./ Т.Н.Смирнова, А.А.Сорокин, Ю.Ф.Тельнов. М.: Финансы и Статистика, 2001.

18. Трофимов С.А. CASE-технологии: практическая работа в Rational Rose/ С.А.Трофимов. М.: М.:ЗАО «Издательство БИНОМ», 2001.
19. Трэйси Мэйор Методологии оценки ИТ/ Мэйор Трэйси// Директор ИС. 2002. № 9 (<http://www.osp.ru/cio/2002/09/056.htm>).
20. Фокс Дж. Программное обеспечение и его разработка/ Дж.Фокс. М.: Мир, 1985.
21. Черемных С.В. Моделирование и анализ систем. IDEF – технологии: практикум/ С.В.Черемных, В.С.Ручкин, И.О.Семенов. М.: Финансы и статистика, 2002.
22. Martin, J. Information Engineering: a trilogy// Englewood Cliff: Prentice-Hall, 1989-1990.
23. Whitten, Jeffrey L. System analysis and design methods. 3rd ed. IRWIN, 1994.

Учебное издание

Братищенко Владимир Владимирович

ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

Учебное пособие

Издается в авторской редакции

Компьютерная верстка Братищенко В.В.

ИД № 06318 от 26.11.01

Подписано в печать 25.06.04 Формат 60×90 1/16. Бумага офсетная. Печать трафаретная. Усл.печ.л. 5,3. Уч.-изд.л. 4,7. Тираж 100 экз.. Заказ _____.

Издательство Байкальского государственного университета
экономики и права.

664003, Иркутск, ул. Ленина, 11.

Отпечатано в ИПО БГУЭП.