

**Современный
Гуманитарный
Университет**

Дистанционное образование

Рабочий учебник

Фамилия, имя, отчество _____

Факультет _____

Номер контракта _____

ПРИНЦИПЫ ПОСТРОЕНИЯ АВТОМАТИЗИРОВАННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ (АИС)

ЮНИТА 3

ТЕХНИЧЕСКОЕ И РАБОЧЕЕ ПРОЕКТИРОВАНИЕ АИС

МОСКВА 2000

Разработано В.Н. Кузубовым

Рекомендовано Министерством
общего и профессионального
образования Российской Федерации
в качестве учебного пособия для
студентов высших учебных заведений

КУРС: ПРИНЦИПЫ ПОСТРОЕНИЯ АВТОМАТИЗИРОВАННЫХ ИНФОРМАЦИОННЫХ СИСТЕМ

Юнита 1. АИС по отраслям применения.

Юнита 2. Предпроектный анализ, разработка технического задания
АИС.

Юнита 3. Техническое и рабочее проектирование АИС.

Юнита 4. Сдача в эксплуатацию и сопровождение АИС.

ЮНИТА 3

Излагаются основные принципы эскизного, технического и рабочего проектирования распределенных автоматизированных информационных систем (АИС). Основное внимание уделено проблемам создания и испытания прототипа АИС, проектирования архитектуры распределенного информационного обеспечения корпоративной АИС, выбору, оценке и интеграции покупных и повторно используемых компонент (ПИК).

Кроме того, рассматриваются вопросы документирования технического и рабочего проектов.

Для студентов Современного Гуманитарного Университета

Юнита соответствует образовательной профессиональной программе
№1

ОГЛАВЛЕНИЕ

| | |
|---|------|
| | стр. |
| ДИДАКТИЧЕСКИЙ ПЛАН | 4 |
| ЛИТЕРАТУРА | 5 |
| ТЕМАТИЧЕСКИЙ ОБЗОР | 7 |
| 1. Техническое проектирование АИС | 7 |
| 1.1. Разработка эскизного проекта | 7 |
| 1.1.1. Разработка исходных моделей корпоративной АИС | 7 |
| 1.1.2. Разработка пользовательских интерфейсов и прототипа АИС | 20 |
| 1.1.3. Испытание прототипа и доработка ТЗ по результатам испытаний | 25 |
| 1.1.4. Функциональная структура АИС промышленной корпорации | 27 |
| 1.2. Разработка архитектуры АИС | 33 |
| 1.2.1. Основные принципы разработки архитектуры АИС | 33 |
| 1.2.2. Основные методы разработки состава и структуры информационного обеспечения | 36 |
| 1.2.3. Выбор критерия оптимального синтеза распределенной архитектуры ИО АИС | 40 |
| 1.2.4. Выбор параметров архитектуры ИО распределенной АИС | 46 |
| 1.3. Разработка спецификаций на функции, физический компонент и программное обеспечение АИС | 53 |
| 1.4. Разработка и документирование ТП в соответствии с ТЗ и стандартами | 54 |
| 2. Рабочее проектирование АИС и интеграция компонентов .. | 59 |
| 2.1. Разработка прикладных программ | 59 |
| 2.1.1. Разработка исходных текстов программ | 59 |
| 2.1.2. Прикладное программирование на базе повторно используемых компонентов | 62 |
| 2.1.3. Трансляция, отладка, тестирование и оценка качества программ и их соответствия требованиям ТЗ | 67 |
| 2.1.4. Документирование прикладных программ | 73 |
| 2.2. Разработка баз данных | 73 |
| 2.2.1. Структуры баз данных | 73 |
| 2.2.2. Ввод информации для контрольных примеров и испытания баз данных | 80 |
| 2.3. Интеграция компонентов АИС | 84 |
| 2.3.1. Интеграция прикладных программ в функциональные блоки .. | 84 |
| 2.3.2. Интеграция баз данных, технических и программных компонентов | 85 |

| | |
|--|----|
| 2.4. Комплексная отладка и тестирование, разработка плана и методик приемо-сдаточных испытаний АИС | 90 |
| ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ | 97 |
| ГЛОССАРИЙ* | |

* Глоссарий расположен в середине учебного пособия и предназначен для самостоятельного заучивания новых понятий.

ДИДАКТИЧЕСКИЙ ПЛАН

Разработка эскизного проекта. Модели корпоративной АИС. Разработка пользовательских интерфейсов и прототипа АИС. Испытание прототипа и доработка ТЗ по результатам испытаний. Функциональная структура АИС промышленной корпорации. Архитектура АИС. Основные принципы разработки архитектуры АИС. Основные методы разработки состава и структуры информационного обеспечения. Критерии оптимального синтеза распределенной архитектуры ИО АИС. Выбор параметров архитектуры ИО распределенной АИС. Разработка спецификаций на функции, физический компонент и программное обеспечение АИС. Разработка и документирование ТП в соответствии с ТЗ и стандартами.

Рабочее проектирование АИС и интеграция компонентов. Разработка прикладных программ. Разработка исходных текстов программ. Прикладное программирование на базе повторно используемых компонентов. Трансляция, отладка, тестирование. Оценка качества программ и их соответствия требованиям ТЗ. Документирование прикладных программ. Разработка баз данных. Структуры баз данных. Ввод информации для контрольных примеров и испытания баз данных. Интеграция компонентов АИС. Интеграция прикладных программ в функциональные блоки. Интеграция баз данных, технических и программных компонентов. Комплексная отладка и тестирование, разработка плана и методик приемо-сдаточных испытаний АИС.

ЛИТЕРАТУРА

Базовая

- *1. Мамиконов А.Г. Проектирование АСУ: Учебник. М., 1987.

Дополнительная

- *2. Макетирование. Проектирование и реализация диалоговых информационных систем. М., 1993.
- *3. Колянов Г.Н. Консалтинг при автоматизации предприятий (подходы, методы, средства). М., 1997.
- *4. Персон Р. Access для Windows 95 в подлиннике. Спб., 1997.

Примечание. Знаком (*) отмечены работы, на основе которых составлен тематический обзор.

1. ТЕХНИЧЕСКОЕ ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ АИС

1.1. Разработка эскизного проекта

1.1.1. Разработка исходных моделей корпоративной АИС

Этап проектирования - это условно выделенная часть процесса проектирования, сводящаяся к выполнению одной или нескольких проектных процедур, общность которых определяется принадлежностью получаемых проектных решений к одному иерархическому уровню описания.

Эскизное проектирование не является обязательным этапом создания АИС и иногда рассматривается как часть технического проектирования. Однако при разработке сложных систем **эскизное проектирование** - предварительное проектирование, описывающее крупноблочную модель структуры АИС на уровне описаний пользовательских интерфейсов и логических моделей данных, - как правило, необходимо.

Структура АИС - это совокупность устойчивых связей, способов взаимодействия элементов системы, определяющая ее целостность и единство, при этом все, что находится в предметной области за границами системы, считается **средой (окружающей) функционирования АИС**.

Для современных корпоративных АИС характерна **распределенная обработка данных**, когда разные функции или подсистемы, интегрируемые в единую, функциональную структуру АИС, реализуются на программно-аппаратных платформах разного типа, территориально удаленных друг от друга. Отдельные части таких АИС часто создавались независимо друг от друга и, в этом случае, важнейшей задачей проекта является **интеграция информационных ресурсов** - создание единого информационного обеспечения АИС из информационных компонентов, реализованных с помощью различных моделей данных, языков программирования и функционирующих в различных операционных системах.

Основные принципы разработки АИС сформировались в результате многочисленных исследований и экспериментов, а также опыта разработчиков. Имеются разные подходы к процессу проектирования,

* Жирным шрифтом выделены новые понятия, которые необходимо усвоить. Знание этих понятий будет проверяться при тестировании.

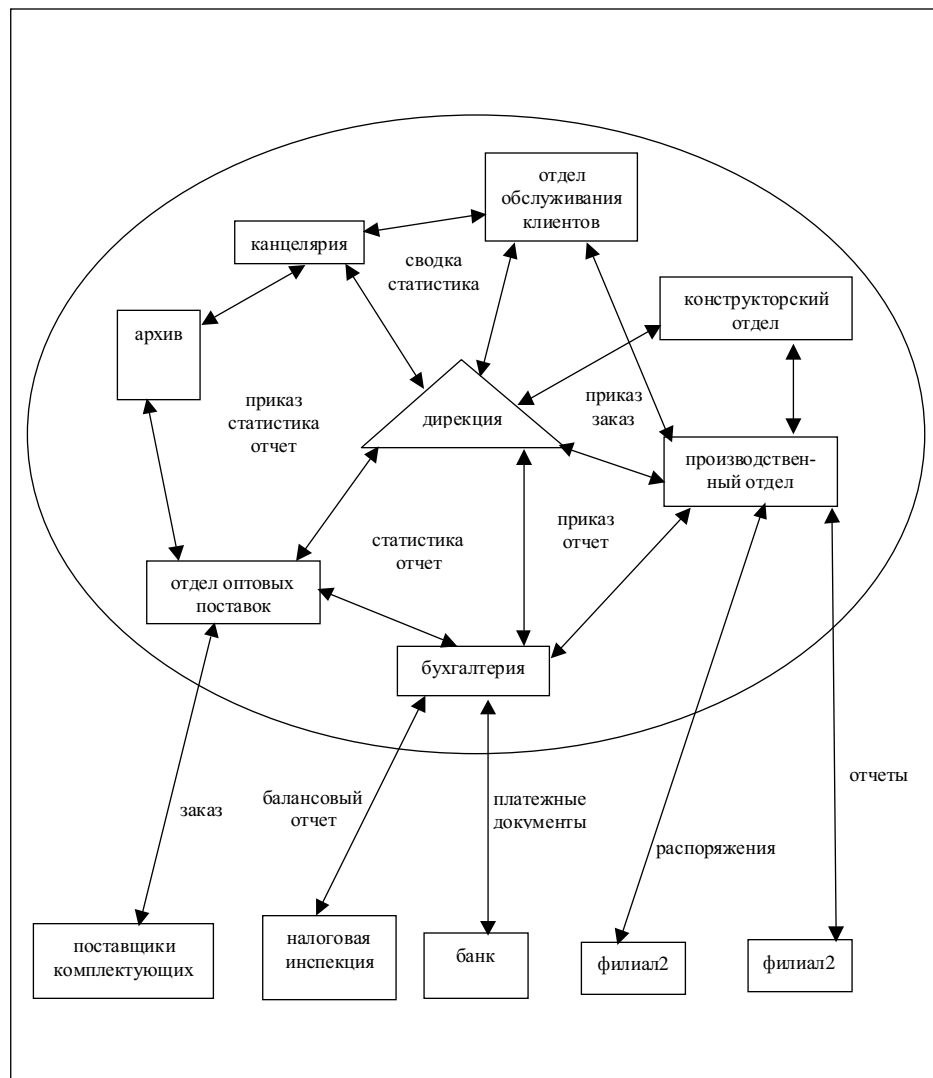


Рис.1. Обобщенная схема внутренних и внешних связей корпорации

но основные отличия между ними заключаются в использовании различной терминологии и приоритетов. Мы будем рассматривать подход, при котором приоритеты отдаются целям создания АИС.

Это значит, что АИС создается не ради использования самых последних версий компонентов, интегрированных в сверхсовременной (с технической точки зрения) системе, а ради эффективного достижения целей объекта, для которого она создается.

С точки зрения системного аналитика, все организации весьма похожи друг на друга. В структуру каждой из них, независимо от рода деятельности, входят многочисленные подразделения, непосредственно осуществляющие тот или иной вид деятельности корпорации, а также дирекция, бухгалтерия, канцелярия и т.д.

Подразделения корпорации (см. рис. 1.) пронизаны вертикальными и горизонтальными связями, обмениваются между собой информацией, а также выполняют отдельные функции в общекорпоративной структуре. Некоторые из подразделений, например, дирекция, финансовые и снабженческие службы взаимодействуют с внешними партнерами (банк, налоговая инспекция, поставщики и т.д.), а также филиалами самой компании.

Таким образом, *любая организация* - это совокупность взаимодействующих элементов (подразделений), каждый из которых может иметь свою структуру. Элементы связаны между собой функционально, т.е. они выполняют отдельные виды работ в рамках единого процесса, а также информационно, обмениваясь документами, факсами, письменными и устными распоряжениями и т.д. Кроме того, эти элементы взаимодействуют с внешними системами, причем их взаимодействие также может быть как информационным, так и функциональным. И эта ситуация справедлива практически для всех организаций, каким бы видом деятельности они ни занимались - для правительственного учреждения, банка, промышленного предприятия, коммерческой фирмы и т.д.

Такой общий взгляд на организацию позволяет сформулировать некоторые общие подходы к построению корпоративных информационных систем (рис.2.).

Корпоративная АИС должна улучшать технико-экономические показатели деятельности корпорации. Можно выделить следующие группы проблем, которые необходимо учитывать с самого начала проектирования:

- экономико-методические;
- системно-технические;
- организационно-технические.

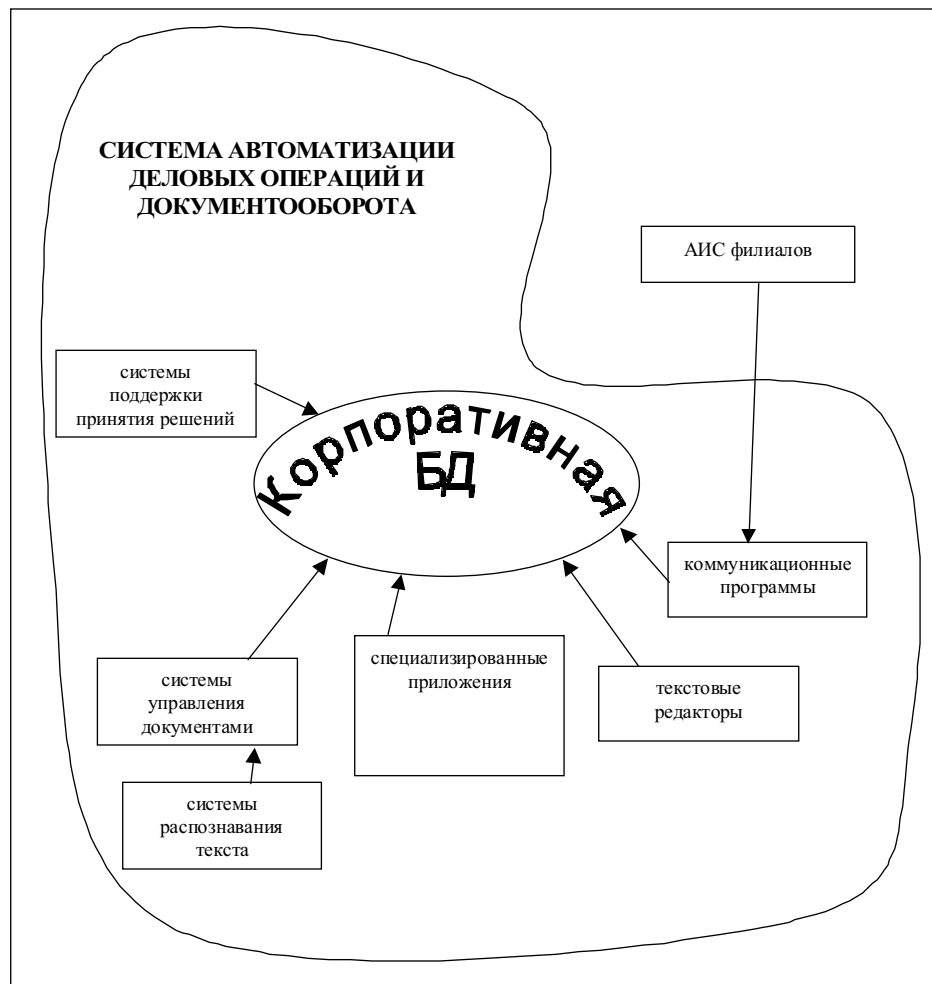


Рис. 2. Обобщенная схема корпоративной информационной системы

Экономико-методические проблемы разработки корпоративных АИС определяют экономическую природу систем данного класса. Они сводятся в основном к следующему.

1. Построение общей экономико-организационной модели, определяющей цели, критерии и функции системы управления корпорацией. Экономико-организационная модель АИС является основой определения взаимосвязи и моделирования задач управления,

которые формируют требования к математическим методам, а также информационному, техническому и программному обеспечению. Она позволяет объединить в интегрированную функциональную систему составные части функциональных подсистем АИС.

2. Выделение первоочередных задач, подлежащих решению с помощью АИС и обеспечивающих наибольший первоначальный эффект. Определение первоочередных задач осуществляется на базе соотношения эффекта, достигаемого в результате повышения качества их решения, и издержек на создание компонентов АИС, обеспечивающих выполнение этих задач.

3. Приведение структуры корпорации в соответствии с требованиями и возможностями методов и средств управления. Выполнение этого принципа означает мобилизацию резервов для повышения результатов деятельности предприятия.

4. Адекватное представление закономерностей деятельности структурных единиц корпорации в модели управления в соответствии с установленными функциями и задачами управления. Это должно обеспечиваться разработкой экономико-математических моделей, а также формализацией процессов в действующей системе управления.

5. Согласование методов воздействия на ход регулирования процессов по выполнению обязательств корпорации перед партнерами. Рассогласование в действии каких-либо методов регулирования может препятствовать эффективному использованию современных методов и средств управления.

Системно-технические проблемы разработки АИС формируются под влиянием требований используемых компонентов АИС. Они сводятся в основном к следующему:

1. Представление процессов решения задач на языке, воспринимаемом как человеком, так и компонентами АИС. Это должно проявляться в том, что в рамках экономико-организационной модели необходимо формализованное (математическое) представление закономерностей деятельности предприятий в соответствии с задачами управления.

2. Установление строгого графика последовательности прохождения информации, используемой для решения задач управления, что определяется свойствами систем обработки данных, используемых в системах управления.

Выполнение этого принципа обеспечивается разработкой специального компонента АИС, строго контролирующего графики решения задач управления и обработки документов. Этот компонент, по сути дела, точно отражает динамические свойства системы управления корпорацией.

3. Организация информационных потоков между подразделениями корпорации и ее деловыми партнерами в локальных и глобальных сетях АИС.

4. Организация централизованных, корпоративных хранилищ данных.

5. Создание правового обеспечения процессов функционирования АИС в ходе решения задач управления деятельностью корпорации, ее внешними связями. Правовое обеспечение заключается в разработке положений о деятельности подразделений корпорации в условиях функционирования АИС, а также должностных инструкций для сотрудников, работающих на автоматизированных рабочих местах (АРМах). При решении проблем правового обеспечения особое внимание должно быть уделено организации эффективного взаимодействия человека и компонентов АИС.

Таким образом, проблемы системно-технического характера накладывают дополнительные ограничения при разработке АИС.

Совокупность экономико-методических и системно-технических проблем определяет *организационно-технические* проблемы разработки АИС. Эти проблемы следующие.

1. Руководство разработкой и внедрением АИС осуществляет высшее управляющее звено корпорации. Действие этого принципа объясняется рядом факторов. Например, решение многих вопросов в процессе разработки и внедрения АИС (установление приоритетных задач управления, введение в действие новых методов информационных работ и т. д.) относится к компетенции только высшего руководства.

2. Рациональная организация управления ходом разработки АИС заключается в создании эффективной структуры коллектива разработчиков и в организации его тесного взаимодействия с персоналом аппарата управления.

3. Моделирование проектных решений на прототипе АИС до ее внедрения. Соблюдение этого принципа проявляется в предварительном испытании предлагаемых в проекте экономико-математических моделей решения задач до начала практического использования их в деятельности аппарата управления. Это позволяет сократить время внедрения АИС и повысить качество проектов. Кроме того, предварительное испытание прототипа содействует повышению квалификации специалистов аппарата управления и сокращению периода их переподготовки.

4. Подготовка персонала аппарата управления предприятия или объединения к работе в условиях использования новых методов и средств должна являться неотъемлемой частью общего комплекса работ

по созданию АИС. Специалисты должны знать возможности и условия использования новых методов и средств. Это достигается путем повышения квалификации персонала корпорации.

Экономико-организационная модель системы управления корпорацией, которая является результатом предпроектного анализа, представляет собой качественное описание организации управления в соответствии с закономерностями и условиями функционирования корпорации в общей системе соответствующего сектора рынка продукции и услуг. Эта модель характеризует такие группы показателей, как:

- цели и критерии функционирования корпорации;

- функции и задачи управления производством, ресурсами и продажами;

- закономерности развития производства и сбыта продукции и услуг;

- требования к методам и средствам решения задач управления;

- требования к организационной структуре корпорации, условиям и особенностям формирования структуры аппарата управления.

Роль экономико-организационной модели в построении АИС заключается в следующем:

- во-первых, экономико-организационная модель позволяет в процессе анализа установить важность тех или иных задач управления в достижении целей развития и функционирования корпорации. В результате этого разработчики и специалисты заказчика получают возможность объективно оценить качество решения тех или иных задач и их место в очередности развития АИС;

- во-вторых, экономико-организационная модель дает возможность точнее установить взаимосвязь задач управления и на основе этого расчленить функциональный компонент АИС на составные части, что позволяет более четко распределить работы между специалистами, а также установить границы и каналы их взаимодействия;

- в-третьих, экономико-организационная модель, определяя взаимосвязь задач управления, позволяет выделить их общее информационное обеспечение. В результате этого разработчики получают возможность организовать эффективную разработку построения информационного компонента АИС (системы обработки данных).

Корпоративные АИС создаются и вводятся в эксплуатацию по частям, поэтому при переходе к вводу в эксплуатацию очередной части возникает необходимость увязки всех задач, решаемых системой. На этой стадии экономико-организационная модель предоставляет возможность разработчикам решать эту проблему без дополнительных обследований и анализа.

Построение экономико-организационной модели осуществляется на этапе эскизного проектирования АИС на основе использования общих принципов системного анализа.

Первой основной фазой системного анализа при построении экономико-организационной модели является определение цели развития и функционирования корпорации. При этом цель понимается как место (реальное и планируемое) корпорации в общем отраслевом комплексе (в соответствующем секторе рынка продукции и услуг).

Как правило, цель развития и функционирования корпорации представляется одним из показателей их деятельности - показателем увеличения объема выпуска и продаж продукции, снижения затрат, увеличения прибыли и т. д. При этом показатель цели должен получить количественную оценку.

В соответствии с целью развития и функционирования корпорации осуществляется анализ направлений совершенствования ее структуры – расширение производства, организация новых филиалов, специализация и кооперирование производственных подразделений и т.д. На основе этого анализа определяются тенденции и направления развития или совершенствования структуры корпорации.

На основе построенной модели развития и совершенствования производства и сбыта должно осуществляться формирование задач управления корпорацией. Следовательно, экономико-организационная модель должна включать как задачи организации текущего управления, так и перспективные задачи, возникающие в связи с тенденциями развития. Формулировка задач управления производится в соответствии с общими положениями организации деятельности корпорации. Степень детализации задач управления определяется квалификацией разработчиков, а также уровнем изученности предприятий данной отрасли.

Для построения экономико-организационной модели корпорации возможно использование двух подходов.

Первый подход сводится в основном к следующему:

формулируются общие цели, функции и задачи управления;

производится последовательная детализация задач управления до непосредственно производственных и трудовых процессов.

Таким образом, последовательная детализация функций должна обеспечить построение своеобразного дерева, или графа, задач управления деятельностью корпорации. В результате этого одновременно обеспечивается и установление взаимосвязи задач.

Такой подход позволяет четко определить перечень задач, их взаимосвязь, а также иерархию. Дерево задач предоставляет

возможность для выявления требований к методам их решения, структуре аппарата управления, а также к информационному обеспечению будущей АИС.

Вместе с тем такой подход обладает и определенными недостатками. Прежде всего, он чреват опасностью упущения каких-либо задач управления. Кроме того, в процессе построения дерева задач вероятно возникновение многих вопросов о содержании отдельных задач.

Второй подход базируется на использовании в качестве основы построения экономико-организационной модели действующей организации системы управления производством.

В основном этот подход сводится к следующему:

анализируется организационная структура действующей системы управления корпорации;

определяются задачи, решаемые отдельными подразделениями органа управления;

в соответствии с организационной структурой разрабатывается дерево задач;

фактическое дерево задач управления дополняется новыми задачами, вытекающими из анализа тенденций развития и совершенствования корпорации.

Таким образом, и этот второй подход обеспечивает построение схемы взаимосвязи задач управления. Практическая реализация этого подхода более легкая. Она осуществляется в процессе предпроектного анализа деятельности корпорации, для осуществления которого разработаны специальные методики.

Однако этот подход также обладает определенными недостатками. Прежде всего, он не обеспечивает точной формулировки задач управления, а отображает сложившуюся практику их выполнения.

Как первый, так и второй подход обладает определенными преимуществами и недостатками. Поэтому они самостоятельно, как правило, не применяются. Обычно на практике используются оба эти подхода в комбинации, т.е. обеспечивается реализация их преимуществ и преодолеваются недостатки.

В процессе построения дерева задач управления должно осуществляться одновременно и *определение целей* их решения. При этом в основу должна быть положена общая цель развития корпорации. Однако это не означает, что она будет присуща всем задачам управления. Дело в том, что нижележащий уровень задач (в иерархическом дереве задач) является средством достижения задач управления вышележащего уровня. Таким образом, показатель средства

достижения цели первого уровня может быть целью второго уровня. Правильное установление целей по уровням задач управления является основой решения проблем организации функционирования управляющей и управляемой систем корпорации.

Одновременно решение проблемы установления целей и критериев решения задач управления должно обеспечить установление централизации и децентрализации прав и обязанностей по уровням органа управления. В результате всех вышеперечисленных операций формируется основа экономико-организационной модели, на базе которой могут быть сформулированы основные задачи функционального компонента в эскизном (техническом, если эскизное проектирование не выполняется) проекте.

Следующей фазой в построении экономико-организационной модели является определение основных черт организационной структуры аппарата управления (напомним, что сотрудники аппарата управления, работающие на АРМах, представляют собой организационный компонент АИС). В качестве основы на этой фазе исследования может быть принята типовая структура управления для корпораций данной отрасли. При этом разработчики должны использовать все достижения передового опыта и науки в области развития и совершенствования организационных структур. Функции каждого подразделения должны определяться в соответствии с разработанным деревом задач управления производством. В результате этого фактически сложившаяся организационная структура может претерпеть изменения как за счет уточнения состава подразделений и их подчиненности, так и в связи с уточнением их функций и задач.

Таким образом, экономико-организационная модель определяет основные контуры будущей автоматизированной системы управления корпорацией. Она обеспечивает последующее проведение анализа, а также разработку эскизного и технического проекта. Вместе с тем она не теряет своего значения и на последующих этапах разработки и развития АИС. При этом на стадии развития АИС, при переходе к ее новым очередям, роль экономико-организационной модели возрастает.

Экономико-организационная модель используется на всех этапах разработки и в процессе развития АИС. Практика показала, что экономико-организационная модель необходима:

- при проведении эскизного проектирования и прототипирования АИС;

- при разработке технического и рабочего проектов;

- в процессе развития АИС при переходе от одной версии к другой.

При проведении испытания на прототипе экономико-организационная модель используется для оценки роли (важности) задач управления в достижении стоящих перед объектом управления целей.

На этапе технического проектирования экономико-организационная модель позволяет увязать задачи управления в информационном и технологическом отношении. Это достигается в результате того, что экономико-организационная модель обеспечивает выявление требований к:

- составу и частоте использования планово-статистических данных в ходе решения стратегических задач управления корпорацией;

- характеру (номенклатуре) постоянных конструкторско-технологических нормативных данных, находящихся применение в процессе решения ряда задач управления производством;

- направлениям распределения данных по подразделениям управления корпорацией;

- методам и способам обработки данных;

- организации согласования АИС с другими системами глобальных сетей.

На этапе рабочего проектирования экономико-организационная модель используется для обеспечения взаимной увязки рабочей документации с остальными частями технического проекта, которые подлежат реализации при последующем развитии АИС. На этом этапе экономико-организационная модель используется:

- для уточнения схем связи программ и данных рабочего проекта со всеми разделами технического проекта;

- при согласовании процессов взаимодействия корпоративной АИС с АИС деловых партнеров, государственных и территориальных органов управления.

Таким образом, экономико-организационная модель на всех этапах разработки и развития АИС сохраняет свое значение и играет важную роль в обеспечении *методического единства* проекта.

Экономико-организационная модель детализируется, в том числе, экономико-математическими методами решения конкретных задач управления. Известно, что экономико-математические методы и модели являются одним из языков представления содержания задач управления процессами деятельности корпорации. Это позволяет моделировать производственные и хозяйственные процессы и выбирать оптимальные варианты их осуществления.

Вместе с тем экономико-математические методы и модели не являются пассивным элементом в системе управления, так как они предъявляют высокие требования к точности выделения и формулировке

задач управления и, следовательно, оказывают влияние на проектирование функционального и информационного компонентов АИС.

Эти модели являются одной из форм представления закономерностей развития и функционирования корпорации с целью исследования их вариантов и выбора наилучших из них для достижения поставленных целей и задач. Возможности математического аппарата и ЭВМ не позволяют осуществить разработку какой-либо единой всеобъемлющей математической модели, отражающей все процессы развития и функционирования корпорации, поэтому экономико-математическое моделирование производится в разрезе отдельных задач управления.

Построение экономико-математических моделей производится по следующим этапам:

- качественная формулировка задачи (комплекса задач);

- точное определение цели, критерия и ограничений решения выделенной задачи управления;

- установление закономерностей рассматриваемого аспекта деятельности корпорации;

- математическая формулировка модели решения задачи управления;

- разработка алгоритма и программы использования экономико-математической модели решения задачи;

- экспериментальное решение задачи с помощью экономико-математической модели и анализ полученных результатов (прототипирование);

- корректировка или уточнение ранее сформулированной экономико-математической модели и дополнительная ее проверка экспериментальными расчетами;

- уточнение требований ТЗ к информационному и организационно-техническому обеспечению АИС.

Качественная формулировка задачи управления, а также точное определение цели, критерия и ограничений ее решения должны осуществляться разработчиками АИС совместно со специалистами заказчика. Все компоненты модели должны получить строгое и однозначное определение.

Формулировка задачи должна быть документально оформлена (задание на разработку прикладных программ) и утверждена разработчиком и заказчиком АИС. Утвержденные формулировки задач являются основой построения экономической модели корпорации (функционального компонента АИС). Они разрабатываются системными аналитиками и специалистами, владеющими методами

математического моделирования. В настоящее время широко используются готовые, апробированные типовые проектные решения (ТПР), которые позволяют разрешать многие проблемы построения экономико-математических моделей. Кроме ТПР, которые должны быть запрограммированы, при построении АИС применяются “повторно используемые компоненты (ПИК)”, представляющие собой программно-информационные компоненты, готовые к использованию (возможно после небольшой настройки параметров). И те, и другие компоненты, перед передачей в эксплуатацию подвергаются проверке на реальных данных корпорации.

В рамках разработанных экономико-организационных и экономико-математических моделей осуществляется подготовка принципиальной схемы информационного обеспечения АИС.

На первых этапах разработок схема информационного обеспечения представляется в укрупненной форме - в виде потоков основных данных, используемых для решения выделенных в экономико-организационной модели задач управления. На этапе рабочего проектирования формируется детальная схема информационного обеспечения.

Укрупненная схема информационного обеспечения позволяет организовать параллельно с дальнейшими работами по проектированию АИС развертывание подготовительных мероприятий. К их числу относятся:

- ревизия действующего в корпорации документооборота;
- выработка направлений его совершенствования.

На этапе рабочего проектирования составляются детальные концептуальные и логические схемы информационного обеспечения. Эти схемы в соответствии с требованиями экономико-математических моделей точно определяют:

- номенклатуру всех данных;
- взаимные связи между данными;
- процедуры обработки данных.

В 90-х годах для разработки и сопровождения экономико-организационной модели объектов управления, для которых создаются АИС, стали широко применяться средства автоматизированного проектирования - CASE-системы. **Назначение CASE-систем** состоит в создании высокопроизводительного окружения, обеспечивающего методы и технологии анализа, разработки и прототипирования сложных АИС на базе экономико-организационной модели объекта управления (в нашем случае, корпорации).

1.1.2. Разработка пользовательских интерфейсов и прототипа АИС

CASE-система предоставляет средства для быстрой разработки прототипа будущей системы. Прототип позволяет своевременно выявить истинные потребности пользователя, обеспечивает обратную связь между разработчиком и пользователем. С помощью прототипа пользователь может оценить приемлемость предлагаемого варианта построения прикладной системы. Каждому участнику разработки дается возможность наглядно представить, как будет выглядеть конечный результат.

Прототип АИС - действующая модель АИС, созданная на базе экономико-организационной модели корпорации средствами CASE-системы на стадии эскизного проектирования. **Прототип АИС предназначен** для анализа, корректировки и согласования с заказчиками решений эскизного проекта и требований ТЗ на действующей модели АИС.

В современных CASE-системах имеются возможности быстрого создания полноценных пользовательских интерфейсов, как правило, с помощью средств языков 4GL. Созданные с помощью этих средств прототипы АИС, с точки зрения пользователя, полностью имитируют реальную рабочую среду соответствующих АРМов.

Фактическое использование экономико-математических моделей должно осуществляться до начала внедрения АИС на этапе разработки и испытания прототипа. Например, модели:

- оптимизации производственной программы;
- исследования временных аспектов сбыта продукции;
- приобретения материалов и комплектующих изделий, –

могут применяться для дальнейшего уточнения требований ТЗ и деталей общей экономико-организационной модели корпорации.

В рамках прототипа проводится первый этап функциональной интеграции экономико-математических моделей задач в единую экономико-организационную модель корпоративной АИС. При этом уточняются требования ТЗ ко всем компонентам АИС. Возникает постоянная обратная связь, когда задачи управления и управляемые объекты (корпорация и ее составные части) определяют содержание и характер экономико-математических моделей. Прогон моделей на прототипе уточняет требования к задачам управления и организационным аспектам деятельности корпорации.

В процессе разработки АИС наблюдается стремление к использованию наиболее проверенных и относительно простых

экономико-математических моделей. Это определяется их требованиями к исходному информационному обеспечению. Кроме того, введение в АИС относительно простых моделей облегчает процесс подготовки персонала к работе в новой системе управления.

Использование экономико-математических моделей носит комплексный взаимоувязанный характер, чтобы результаты одной задачи обеспечивали получение исходных данных для решения других. Таким образом, в процессе разработки возникает интеграция моделей, что обеспечивает непрерывность автоматизированного процесса обработки информации в ходе решения задач управления.

В АИС производственных корпораций используются следующие группы экономико-математических моделей решения задач управления: исследования (маркетинг) секторов рынка, на которых действует корпорация;

- оптимизация производственных программ;

- построение балансов деятельности корпорации (матричные модели);

- оперативное планирование производства продукции;

- регулирование материально-технического снабжения;

- оптимизация эксплуатации оборудования;

- оптимизация графиков выполнения обслуживающих работ (ремонт, профилактики и т. д.);

- оптимизация процессов сбыта продукции и использования транспортных средств;

- комплексные аналитические расчеты результатов деятельности;

- стратегическое планирование.

Еще одной проблемой при разработке прототипа является проблема создания эффективных пользовательских интерфейсов для АРМов.

Проектирование интерфейсов пользователей современных АИС состоит в использовании интегрированного набора средств, помогающих разработчику в создании и управлении различными интерфейсами пользователей. При этом ставится задача отделить процесс создания интерфейса пользователей от разработки прикладных программ, которые не должны связываться с конечными пользователями напрямую. Основные особенности современного интерфейса с пользователями состоят в следующем:

- наличие механизмов управления окнами;

- использование готовых графических символов (икон) для отображения управляемых объектов;

непосредственное манипулирование графическими объектами и окнами посредством “мыши”;

объектно-ориентированное проектирование диалоговых систем.

Для реализации интерфейсов создаются и используются библиотеки технологических интерактивных программ, позволяющих использовать устройства ввода команд управления и графических элементов при наличии обратной связи, отображающей на дисплее результаты манипулирования ими. Для этого разрабатываются методы и языки проектирования интерфейсов пользователей, которые должны соответствовать проблемной области информационной системы. Эти языки можно отнести к компонентам языков четвертого поколения (4GL).

Разработка систем построения и управления интерфейсами пользователей ведется многими фирмами с учетом стандартов открытых систем и постепенной выработкой ряда стандартов де-факто. Графический пользовательский интерфейс (Graphical User Interfaces - GUI) позволяет пользователю АРМа легко и удобно использовать возможности, предоставляемые АИС для выполнения его функций.

Он позволяет решать сложные задачи с помощью манипулирования окнами и элементами изображения (иконами) пользовательских интерфейсов. Основой GUI являются:

наборы графических элементов;

допустимые операции над ними;

меню и области окон для прокрутки изображений.

Задача разработчика сводится к тому, чтобы создать с помощью этих элементов удобную и эффективную среду для сотрудников корпорации, выполняющих свои профессиональные функции.

Одна из наиболее популярных инструментальных систем создания графических пользовательских интерфейсов с вычислительными средствами - это система X Window, которая позволяет формировать основные окна. Это прозрачная относительно сети, независимая от аппаратной платформы, многопроцессная оконная графическая система. Она представляет широкий набор средств для создания окон и рисования в них, но не навязывает жесткого способа общения пользователя с компьютером.

Система позволяет организовать на экранах дисплеев иерархию перекрывающихся окон, манипулировать их размерами и положением. Она поддерживает применение набора примитивов рисования в окнах и обеспечивает вывод высококачественного текста с различными шрифтами. Она не формирует графические образы, а только обеспечивает функционирование различных систем моделирования изображения.

Современное развитие компьютерной графики идет в направлении *мультимедиа*. Для нее характерно использование средств отображения высокого разрешения на больших дисплеях с особыми методами формирования, хранения и обработки изображений. В системах мультимедиа накапливаются, обрабатываются и представляются пользователям сложные образы различных классов (аудио, видео, графика, тексты).

Разработчик интерфейса, хорошо знакомый с организацией работ на АРМах, должен уметь учитывать требования к удобству в работе совершенно незнакомых ему пользователей. Он должен не только удовлетворить их требования с точки зрения вычислительных задач, но и создать интерфейс, удобный с точки зрения физических и психологических потребностей пользователя. Много усилий затрачивается на проектирование интерфейса между человеком и компьютером. Использование цвета, звука, графики - это не случайное, а сознательно принятое решение разработчика. Интерфейс должен включать в себя элемент адаптации, поддерживая интерес пользователя к общению с АИС - по мере того как пользователь привыкает к АРМу, его квалификация возрастает. Интерфейс может меняться, например, увеличивается интенсивность обработки данных. Разработчики должны хорошо разбираться в возможностях аппаратных и программных средств, а также обладать хорошим воображением и проявлять изобретательность при реализации задач АИС.

В основе оригинальных решений по созданию интерфейсов пользователей лежат следующие общие принципы.

1. Интерфейс должен выделяться в качестве отдельного компонента системы. Так же как структуры данных в БД АИС изолируют от прикладных программ обработки этих данных, интерфейс, до определенной степени, необходимо отделить от прикладной задачи.

Интерфейс необходимо проектировать отдельно, как и отдельно разрабатывать базу данных. Состав и форма представления входных и выходных данных должны стать предметом тщательного анализа разработчиков интерфейса.

2. Разработчики интерфейса должны тщательно анализировать возможности и ограничения аппаратных и программных средств АРМа.

3. В процессе разработки требуется новизна, причем необходимо следить за тем, чтобы эта новизна не растворилась в мелочах. Многие пользователи имеют доступ к разным системам, и вряд ли они будут менять свои приемы работы при смене систем. Руководство разработчиков должно способствовать введению стандартов, в которых содержатся рекомендации по разработке интерфейса и использование

библиотек стандартных модулей, которые используются для разработки программных интерфейсов различных систем.

4. Существует большое число общепринятых в эргономике рекомендаций, которые нужно использовать при разработке и организации АРМов в АИС. В то же время форма представления информации на экране не одинакова для различных систем. Графический дизайн зависит от:

- распределения информации на экране;

- словарного состава предложений;

- способа выделения ключевых элементов представления данных и т.д. Разработчики должны знать эти принципы и использовать их при разработке интерфейса.

5. Разработчик интерфейса должен, в первую очередь, понимать и оценивать действия пользователя, направленные на достижение цели решения задачи, а также особенности потенциальных пользователей системы. Нетрудно предположить, что если бы программистам пришлось самим поработать с интерфейсом прикладной задачи целый день, как это делают пользователи, то они бы улучшили интерфейс.

6. Разработчик должен глубоко чувствовать психологические особенности потенциальных пользователей. Полезный способ - это поставить себя на место пользователя, работающего на конкретном АРМе, посмотреть, как на самом деле пользователь взаимодействует с системой в нормальных рабочих условиях. Вряд ли можно ожидать, что пользователь правильно оценит достоинства организации данных по образцам на бумаге, представляемым в формате расположения данных на экране дисплея. Особенности организации ввода трудно понять из описания, пользователь должен нажимать соответствующие клавиши в ответ на сообщения системы. Именно эту роль выполняют прототипы АРМов. На них проходит интерактивный циклический процесс, приводящий к разработке опытных образцов интерфейсов, с которыми работают пользователи и которые изменяются в соответствии с их реакцией до тех пор, пока не будет создан приемлемый продукт.

7. Должны быть предусмотрены средства настройки интерфейса. Хотя общие принципы определяют основу создания интерфейса, они не могут удовлетворять любого пользователя. Даже если условия задачи остаются практически постоянными, потребности пользователей меняются. Правильно спроектированный интерфейс должен быть настраиваемым на нужды разных пользователей, а также на одного пользователя в разные периоды его работы.

1.1.3. Испытание прототипа и доработка ТЗ по результатам испытаний

Практический опыт показывает желательность как можно более полной автоматизации стадий жизненного цикла компонентов АИС. Высокая стоимость разработок автоматизированных информационных систем и их принципиальное значение для успеха организации требует, чтобы каждое проектное решение было оценено с позиций производительности системы, ее качества и эффективности. Для этого прототип, созданный на этапе эскизного проектирования, подвергается тщательным испытаниям. **Испытания прототипа** - это совместное с заказчиками проведение имитационных прогонов АИС на базе прототипа с целью уточнения требований ТЗ и решений эскизного проекта.

Имитация на прототипе - это метод проведения экспериментов с помощью прототипа АИС, моделирующий основные моменты реальной эксплуатации АИС в режимах:

статической имитации - имитационного эксперимента, позволяющего моделировать поведение АИС в определенный момент времени;

динамической имитации - имитационного эксперимента, позволяющего моделировать поведение АИС в течение продолжительного периода времени.

Имитация на прототипе позволяет:

- узнать мнение будущего пользователя о внешнем интерфейсе;
- убедиться, что система может реализовать поставленные перед ней цели;
- ощутить пользователю эффект работы с реальной системой;
- уточнить требования ТЗ, чтобы повысить качество и эффективность использования АИС;

выявить и устранить заранее недостатки системы, с точки зрения пользователя, еще до того как будут сделаны значительные вложения ресурсов в проект;

использовать эффективные средства для уточнения, проверки и документирования требований к информационной системе на начальных этапах проектирования.

CASE-системы обеспечивают создание прототипа системы за счет предоставления возможностей для наглядного проектирования образов экранов, меню, отчетов и форм для ввода данных. При этом можно разрабатывать и связывать с соответствующими шагами диалога тексты подсказок, которые будут доступны пользователю в интерактивном

режиме. И, наконец, могут быть описаны сценарии диалога, определяющие порядок и условия смены состояний системы. Состояния системы связываются с созданными образами экранов. После составления сценария диалога прототип системы может быть запущен на исполнение. Работа прототипа будет имитировать функционирование будущей системы. Пользователь получает возможность оценить работу системы и выдать предложения по ее совершенствованию.

Поддержка проектирования образов экранов, меню, отчетов и форм - главная функция CASE-системы. Эти образы отражают содержание потоков данных и управления реальной АИС. Каждый образ экрана может быть связан с соответствующим потоком данных, который был ранее определен на графической модели и содержится в репозитории (проектная база данных CASE-системы). Если поток данных был определен заранее, то проектирование соответствующего образа экрана сводится к простой задаче его "рисования" с использованием известных компонентов.

Данные и их размеры задаются с помощью выбора нужных значений из всплывающих меню. Ранее определенные значения типов и размеров данных обеспечивают адекватность описания полей экранов. Если характеристики данных задаются в ходе разработки прототипа, то эти данные автоматически пополняют репозиторий. Список ранее определенных элементов и структур данных всегда доступен разработчику прототипа. Возможность присвоения имен образам экранов, полям экранов (вместе с графическими элементами оформления окон) дополняет список средств, необходимых для создания образов экранов. Все элементы пользовательского интерфейса, могут быть передвинуты, скопированы, удалены и видоизменены, после того как они попадут в образ экрана. При этом могут быть изменены атрибуты элементов экрана, правила, определяющие допустимые значения вводимых данных. Тексты подсказок, связанные с элементами пользовательского интерфейса могут быть распечатаны в форме документа. Все эти работы могут выполняться разработчиком и пользователем совместно, непосредственно в ходе испытаний прототипа, что резко повышает скорость и качество проектирования.

Чтобы приблизить общий вид диалога к реальным условиям, цвета элементов образов экранов и назначение используемых клавиш могут быть переопределены. Могут быть переопределены клавиши команд переходов между состояниями системы, команды редактирования полей ввода данных, команды выдачи подсказок, перемещения

курсора и многое другое. Многообразные варианты определения клавиатуры и вида экранов могут быть запомнены и вызваны по имени для использования в прототипе, а затем и в действующей АИС.

Второй важной функцией CASE-системы является поддержка разработки сценария диалога, определяющего порядок смены образов экранов при работе прототипа (допустимые переходы и условия переходов). Разработка сценария диалога включает в себя следующие действия:

- определение возможных состояний системы и присвоение им имен, по которым к ним можно обращаться;

- проектирование образов экранов и их привязка к состояниям системы (при этом может быть создано множество вариантов прототипа системы с использованием имеющихся образов экранов);

- задание нужного варианта настройки образов экранов для всех состояний системы;

- задание начального и конечного состояния для прототипа;

- определение правил редактирования полей ввода данных, влияющих на порядок смены состояний системы;

- запоминание разработанного сценария диалога в репозитории (с возможностью его повторного использования, модификации или включения в состав программных спецификаций).

Третьей важной функцией CASE-системы является обеспечение исполнения прототипа системы в соответствии с заданным сценарием. Из начального состояния системы пользователь может попасть в любое из определенных сценарием допустимых состояний системы. Могут быть заданы различные варианты определения клавиатуры и цветовые характеристики экранов. Таким образом, имеется возможность “проиграть” различные варианты построения пользовательского интерфейса.

После того, как будут проведены полномасштабные испытания на прототипе и их результаты будут согласованы с заказчиками (пользователями), разработчики переходят к доработке технического задания. Повторяется процесс, который описывается в п. 2.7. юниты 2.

1.1.4. Функциональная структура АИС промышленной корпорации

Функциональная структура интегрированной АИС современной промышленной корпорации, которая целиком направлена на использование последних достижений в области информационных технологий, состоит из следующих подсистем:

- системы управления корпорацией;
- системы управления производством (предприятием – структурным подразделением корпорации);
- гибких производственных систем;
- систем автоматизированного проектирования изделий.

Задачи системы управления корпорацией в целом были рассмотрены в юните 2 и в предшествующих подразделах данной юниты, поэтому далее мы остановимся на трех следующих подсистемах.

Автоматизированная система управления производством состоит из двух иерархически связанных подсистем:

- автоматизированной информационной системы управления предприятием (АИСУП), обеспечивающей управление производственной деятельностью на уровне предприятия в целом;
- автоматизированной системы управления гибким автоматизированным участком (ГАУ) АСУ ГАУ.

АИСУП включает в себя следующие функциональные подсистемы:

- технико–экономического развития и маркетинга;
- планирования и оперативного управления производством;
- управления финансовыми ресурсами и бухгалтерского учета;
- управления качеством продукции.

АСУ ГАУ в свою очередь включает в себя подсистемы управления технологическими и организационными процессами.

АИСУП предназначена для управления производственно–экономической деятельностью предприятия и обеспечивает:

- позаказный метод выпуска продукции, позволяющий наиболее полно учесть требования конкретных потребителей;
- стабильность качества изготавливаемой продукции;
- сокращение производственного цикла от момента получения заказа до момента отгрузки готовой продукции;
- сокращение объема незавершенного производства и, соответственно, сокращение потребностей в складских площадях.

Основными методами достижения этих целей являются:

- внедрение между производственными подразделениями экономических отношений, стимулирующих эффективное использование оборудования, материалов и соблюдение производственной дисциплины;

- децентрализация управления и организация горизонтальных связей между производственными подразделениями, обеспечивающие гибкое принятие решений на минимально необходимом уровне иерархии управления;

создание единой транспортно–складской инфраструктуры предприятия, обеспечивающей быстрое перемещение предметов и средств труда, а также достоверность информации о местонахождении и состоянии грузов;

индивидуализация информации о предметах и средствах труда (параметрах конкретных инструментов, оснастки, заготовок, полуфабрикатов, готовых узлов и деталей);

использование безбумажной технологии и общих (централизованных и распределенных) баз данных для увеличения достоверности и актуальности информации и повышения обоснованности управленческих решений.

Областью применения являются машиностроительные и иные предприятия, имеющие различную серийность выпуска продукции (от мелкосерийного до массового).

Результаты решения задач АИСУП являются входными данными для гибких производственных систем (ГПС).

Гибкие производственные системы – это интегрированное, автоматизированное производство нового поколения, обеспечивающее выпуск товарной продукции в условиях изменяющегося рыночного спроса и функционирующее при ограниченном числе работающих. ГПС реализуется в виде гибких автоматизированных участков (ГАУ), линий (ГАЛ), цехов (ГАЦ), заводов (АЗ).

ГПС строятся в соответствии с достижениями научно-технического прогресса путем сочетания передовой техники, технологии производства и управления с высокой квалификацией персонала и призваны обеспечить структурную перестройку мировой экономики. Для них характерны полная автоматизация основных технологических процессов, интеллектуальной деятельности персонала, а также технической подготовки производства.

ГПС предназначены для многономенклатурного автоматизированного производства отдельных деталей и узлов, а также изделий (машин) в машиностроении, производстве товаров потребления (швейных изделий, обуви, холодильников, стиральных машин, шин и т.д.) и в других сферах производства. Использование ГПС эффективно и позволяет повысить отдачу технологических процессов в 2-4 раза за счет:

использования безлюдной и малолюдной технологии;

повышения коэффициента загрузки станков с 0,4-0,6 до 0,85-0,9;

повышения коэффициента сменности с 1,3-1,6 до 2,5-3,0;

сокращения цикла создания и производства новых изделий в 2-3 раза;

сокращения длительность производственного цикла обработки “условного изделия” в 6-10 раз;

применения 0,5-1-месячной программы вместо 3-6-месячной.

ГПС могут применяться в массовом, серийном, мелкосерийном и единичном производстве изделий.

Применение ГПС, как принципиально новых видов организации производства и оборудования, вызвано усиливающейся индивидуализацией запросов потребителей на различные изделия, ужесточением требований к качеству продукции, расширением производства сложной наукоемкой продукции в условиях изменяющегося рыночного спроса и минимального участия персонала. Это направление признано ведущим во всем мире вследствие:

организации комплектного выпуска продукции с максимальным использованием оборудования, трудовых ресурсов и производственных площадей;

возможности периодической смены и обновления продукции с минимальными затратами средств и времени.

Компьютерное управление производством в гибких системах минимизирует степень непосредственного участия операторов в технологических процессах. Это позволяет исключить субъективизм в управлении производством и снизить возможность загрязнения окружающей среды.

ГПС – относительно новое дорогостоящее оборудование, появившееся в конце 70-х годов. В настоящее время, в промышленно развитых странах внедрение ГПС, в том числе АЗ, ведется ускоряющимися темпами. Количество действующих ГПС составляет более тысячи, из них несколько десятков соответствуют понятию АЗ (30 и более единиц технологического оборудования).

Функциональная интеграция производственных АИС характеризуется полной автоматизацией жизненного цикла изделий, которые выпускаются соответствующими предприятиями, поэтому системы автоматизированного проектирования (САПР) являются их неотъемлемой частью.

Современные системы автоматизированного проектирования включают следующие основные составляющие:

средства взаимодействия пользователя с системой, включая языковые и интерактивные;

средства моделирования геометрических форм изделия и изготовления конструкторско-технологической документации;

средства управления базами данных проектируемых изделий, технологической подготовки производства, формирования управляющей

информации для автоматизированных технологических линий и станков.

САПРы позволяют автоматизировать процесс проектирования и изготовления конструкторско-технологической документации и собственно изделия. Они обеспечивают скорость получения конечного продукта, в десятки раз превышающую традиционные методы, более высокое качество продукции и дают возможность перейти к параллельному проектированию и производству без использования традиционной проектной технической документации на бумажной основе.

Системы автоматизированного проектирования должны существенно повысить производительность труда исследователей и разработчиков самых различных корпораций: в машиностроении и радиоэлектронике, строительстве, архитектуре, картографии и других областях.

Традиционными ручными методами проектирования практически невозможно решать целый ряд задач современного производства. Хотя затраты на закупку интеллектуальных систем автоматизированного проектирования достаточно высоки и, кроме того, сопряжены с необходимостью вложения дополнительных средств при построении объектно-ориентированных систем и непосредственной эксплуатации, их широкое использование экономически оправдано.

В настоящее время на мировом рынке известно большое количество различных систем автоматизированного проектирования, от сравнительно простых чертежных систем до интегрированных систем, допускающих параллельное проектирование и учитывающих все технологические циклы подготовки проекта, изготовления и испытания изделий.

Крупнейшие корпорации-производители программного обеспечения развивают использование средств автоматизированного проектирования для разработки проблемно-ориентированных программных систем. Основные направления исследований в данной области включают создание:

- графических станций, аппаратно реализующих наиболее ресурсоемкие алгоритмы проектирования и отображения,

- локальных и глобальных сетей параллельного проектирования и соответствующего программного обеспечения,

- технологии построения программного обеспечения систем проектирования, которая позволяет получать сложные системы из сравнительно простых составляющих,

- систем на основе международных стандартов организации, структуры данных и интерфейсов.

В перспективе, планируется реализация интегрированных инструментальных средств быстрого создания и адаптации проблемно-ориентированных программных систем, основанных на единых методах и языках описания всех элементов производственной деятельности предприятия.

Во многих промышленных САПР (CAD/CAM английская аббревиатура) ярко выражены тенденции к:

- расширению возможностей по проектированию изделий, деталей и узлов сложной пространственной формы;

- моделированию готовых изделий и созданию их реалистических изображений;

- подробному отображению процессов моделирования обработки изделий, не уступающему по качеству видеофильму.

“Средняя” современная САПР обеспечивает:

- создание трехмерной модели детали;

- широкие возможности модификации формы проектируемого объекта;

- реалистическое отображение проектируемого объекта в процессе дизайнерской проработки путем задания свойств предполагаемого материала, его характеристик (например, прозрачности или коэффициента отражения) и условий освещенности;

- расчет и визуализацию траектории инструмента, моделирование обработки и другие функции.

Необходимо отметить, что в функциональные компоненты современных корпоративных АИС все чаще включают подсистемы *компьютерного моделирования процессов управления и развития корпорации.*

Компьютерные системы моделирования дают возможности:

- всесторонне анализировать сложные процессы, ситуации и проекты;

- перебирать множество вариантов и в результате синтезировать рациональные решения;

- ускорять процесс исследований и разработок, при этом сокращаются затраты на реализацию проектов.

Во многих случаях и в технике, и в экономике при оценке эффективности решений невозможно провести натурные эксперименты, поэтому вычислительные и имитационные действия с моделями сложных объектов на базе компьютерных систем обработки информации оказываются единственно возможным инструментом исследований последствий решений и оценки эффективности проектов.

1.2. Разработка архитектуры АИС

1.2.1. Основные принципы разработки архитектуры АИС

Архитектура АИС – это совокупность принципов логической и физической взаимосвязи компонентов АИС.

При создании сложных, распределенных информационных систем, формировании их архитектуры, выборе компонентов и их связей следует учитывать ряд следующих требований:

- архитектура системы должна соответствовать текущим и перспективным целям и стратегическим функциональным задачам создаваемой информационной системы;
- в структуре и компонентах следует предусматривать обеспечение максимально возможной сохранности инвестиций в аппаратные и программные средства, а также в базы данных при длительном развитии, сопровождении и модернизации информационной системы;
- для обеспечения перспективы развития системы следует предусматривать возможность интеграции разнородных вычислительных компонентов и переносимость приложений на различные аппаратные и операционные платформы на основе концепции и стандартов открытых систем;
- архитектура системы должна быть достаточно гибкой и допускать относительно простое, без коренных структурных изменений, развитие и наращивание функций и ресурсов информационной системы в соответствии с расширением сфер и задач ее применения;
- необходимо обеспечить эффективное использование ресурсов информационной системы и минимизировать интегральные затраты на обработку данных в типовых режимах ее функционирования с учетом текущих эксплуатационных затрат и капитальных вложений в создание АИС;
- должны быть обеспечены отказоустойчивость системы и надежная защита данных от ошибок, от разрушения или потери информации, а также авторизация пользователей, управление рабочей загрузкой, резервированием и восстановлением функционирования АИС;
- следует обеспечить комфортный, максимально упрощенный доступ конечных пользователей к управлению и результатам функционирования информационной системы на основе современных графических средств и пользовательских интерфейсов.

Значительная часть приведенных требований обусловлена гибкостью и естественной динамикой интенсивного развития современных информационных технологий и систем. Высокие темпы

роста основных ресурсов аппаратных средств и сохраняющаяся потребность в их увеличении со стороны пользователей приводят к необходимости соответствующего совершенствования технологий создания программных средств и баз данных.

Практически все корпоративные АИС имеют в своем составе следующие компоненты:

физический компонент – материальная основа носителя информационной системы, т.е. средства вычислительной техники и связи, а также периферийные устройства различного назначения;

информационный компонент - организованная определенным образом система данных и знаний (информационное обеспечение) вместе с системой процедур управления, обновления, поиска и завершающей обработки данных;

программный компонент - общесистемное программное обеспечение;

функциональный компонент - система приложений для пользователей АИС, использующая ресурсы трех вышеперечисленных компонент для решения корпоративных задач;

организационный компонент - система нормативной документации, в которой отображены организационные и правовые аспекты функционирования АИС.

АИС и ее компоненты могут быть сосредоточены в одном месте - локальная система; если взаимодействие между компонентами АИС (или между частями одного компонента) происходит посредством каналов связи, то такая АИС называется распределенной. Распределенная, корпоративная АИС - это несколько локальных АИС географически распределенных подразделений корпорации, функционально интегрированных единой экономико-организационной моделью корпорации.

Основным ресурсом любой АИС является информация, поэтому разработку архитектуры АИС целесообразно начинать с разработки состава и структуры информационного обеспечения. Разработка структуры остальных компонентов должна быть направлена на эффективное использование информационных ресурсов, ключевыми частями которой являются:

корпоративные базы данных и знаний;

управление электронными документами и документооборотом;

локальные базы данных;

базы данных рабочих групп и отдельных пользователей.

При разработке структуры компонентов и АИС в целом, необходимо сформулировать критерии ее формирования. В зависимости от

особенностей проблемной области критериями при выборе архитектуры АИС могут быть:

- эффективное использование памяти или производительности программно-технического комплекса;
- трудоемкость или длительность разработки;
- надежность и безопасность.

Важнейшими критериями являются возможность развития АИС и обеспечение возможности изменения состава и функций компонентов с сохранением принципов структурного построения и качества программ и баз данных. В соответствии с целями и задачами проектирования на стадии формирования архитектуры АИС необходимо ранжировать и выделять доминирующие критерии.

Гибкость, модификация и технологическая безопасность при развитии АИС обеспечиваются рядом принципов и правил структурного построения АИС и ее компонентов, а также взаимодействия между ними. Эти правила направлены на стандартизацию и унификацию структуры и взаимодействия компонентов разного ранга и назначения в пределах проблемной области. Часть принципов и правил имеет общий характер и может применяться практически всегда. Другая часть отражает проблемную ориентированность класса АИС и подлежит отработке для эффективного применения в соответствующей области. Основные принципы и правила можно объединить в группы, которые отражают:

- стандартизированную структуру программ или БД определенного класса;

- унифицированные правила структурного построения прикладных программных компонентов;

- правила структурного построения информационных модулей базы данных;

- правила структурного построения межмодульного интерфейса прикладных программ;

- правила внешнего интерфейса и взаимодействия компонентов прикладных программ и БД с внешней средой, с операционной системой и другими типовыми средствами организации вычислительного процесса и контроля.

Таким образом, для обеспечения эффективной разработки необходимо формулировать и соблюдать ряд принципов и правил структурного построения АИС. Эти принципы и правила могут иметь особенности в различных проблемно-ориентированных областях. Однако их формализация и выполнение обеспечивает значительный эффект в снижении трудоемкости и длительности разработки АИС.

Далее эти вопросы рассматриваются подробно на примере разработки архитектуры информационного обеспечения АИС.

1.2.2. Основные методы разработки состава и структуры информационного обеспечения

Информационное обеспечение (ИО) корпоративной АИС будем рассматривать как территориально распределенную систему информационно-взаимосвязанных и определенным образом взаимодействующих баз данных (БД), где каждая БД рассматривается как локальный объект, имеющий определенное информационное содержание и структуру. Таким образом, ИО является рассредоточенной системой памяти соответствующей АИС, проектирование которой порождает целый ряд сложных проблем.

Общая методология создания эффективных структур распределенных баз данных включает решение следующих вопросов:

- общий критерий синтеза структуры базы данных;
- степень централизации информации;
- топология связей в распределенной системе;
- методы управления распределенными данными.

При разработке архитектуры ИО, прежде всего, решаются вопросы: определения числа БД;

их географического размещения;

структуры связей между БД;

общей стратегии поиска информации с целью использования и обновления.

В настоящее время проблемы проектирования распределенных БД рассматриваются как относительно самостоятельные при разработке соответствующей АИСУ. В то же время информационная структура влияет на архитектуру корпоративной сети. Поэтому решение проблемы синтеза оптимальной структуры ИО АИС дает возможность оценить целесообразность существующей или проектируемой программно-технической архитектуры. При разработке информационной структуры территориально распределенных корпоративных БД можно выделить три этапа:

1) проектирование АИС “с нуля”, когда накопленные информационные ресурсы в системе малы. В этом случае проблема выбора оптимальной структуры БД ставится как глобальная стратегическая задача синтеза архитектуры ИО АИС и размещения отдельных БД в интересах всех пользователей системы. На этом этапе возможно достижение максимального эффекта при проектировании за

счет варьирования точек размещения отдельных БД, выбора оптимальной топологии связей между ними и определения оптимальной стратегии их информационного взаимодействия;

2) уже существуют достаточно развитые АИС подразделений корпорации, но размещение необходимых информационных ресурсов по существующим БД существенно отличается от оптимального размещения, а топология связей отличается от оптимальной. В этом случае, задача выбора оптимальной информационной архитектуры ставится как задача:

- размещения дополнительного числа БД;
- выделения части информационных ресурсов из существующих БД для их перераспределения;
- изменения структуры связей между БД с целью оптимизации общей целевой функции.

На этом этапе возможно выделение новых БД и разделение (разбиение) на части уже существующих;

3) формирование географии и структуры связей корпоративных БД в основном уже закончено и возникновение новой структурной единицы (БД) маловероятно. На этом этапе осуществляется лишь вторичная оптимизация за счет перераспределения данных и частичного изменения структуры связей, т. е. изменяется внутренняя архитектура БД. Необходимость перераспределения данных и изменения топологии связей может возникнуть в связи с изменением:

- размещения пользователей системы;
- размещения источников информации;
- характера использования данных (частоты решения задач, периодичности корректировки информации и т. д.).

Решение проблемы выбора оптимальной информационной архитектуры на третьем этапе целесообразно с точки зрения моделирования “идеальной системы” для объективной оценки эффективности функционирования существующей АИС и определения направлений ее дальнейшего развития.

Таким образом, проблема синтеза информационной архитектуры ИО АИС стоит на всех этапах проектирования систем. Решение этой проблемы позволяет разработчикам получить “генеральный план” создания и развития АИС на длительный период.

Анализ практики проектирования территориально распределенных БД позволяет сформулировать следующие методические принципы их проектирования.

1. Комплексный подход к проектированию, учет целей и задач системы в целом при проектировании ее компонентов. В данном случае

имеется в виду определение места, функций и задач БД в соответствующей АИС и анализ влияния параметров системы на выбор структуры корпоративных БД.

2. Синтез оптимальной логической структуры в целях обеспечения эффективной обработки совокупности запросов всех пользователей, возможен только при учете связанности данных в ИО АИС. Например, хотя в процессе синтеза структуры БД общая логическая структура будет подвергаться разбиению на блоки (фрагменты базы данных), это разбиение условно и связано с необходимостью понижения размерностей задач размещения данных в БД.

3. Принцип допущения дублирования информации для территориально распределенных БД имеет существенное значение в связи с тем, что их эффективность зависит от решения одного из основных вопросов, хранить ли необходимую для группы пользователей определенную БД информацию в данной локальной БД или получать ее из других БД? Ответом на этот вопрос является разумный компромисс, когда часть данных приходится дублировать в соответствующей локальной БД, а часть получать из других БД.

4. Как правило, создание любой системы разбивается на ряд общепринятых этапов. Причем на первоначальных этапах выбрать точно оптимальную структуру системы и определить все параметры ее компонент невозможно из-за:

ограниченности сроков на сбор, обработку исходных данных и проведение необходимых расчетов;

отсутствия однозначности решения вопросов выбора целей, задач и функций системы.

В связи с этим улучшение точности и углубление проектных решений носит поэтапный характер согласно стадиям проектирования. Процесс проектирования ИО АИС также разбивается на ряд логически и информационно взаимосвязанных этапов, причем на каждом этапе происходит уточнение общей структуры и отдельных параметров за счет использования дополнительных исходных данных и проведения определенных расчетов.

5. Необходимо формализовать и автоматизировать проектирование. Создание архитектуры распределенных БД связано с проведением большого объема работ по сбору, описанию и обработке исходных данных, по расчету различных вариантов структур. В связи с этим сокращение сроков, улучшение качества проектирования, повышение эффективности системы управления в целом зависят во многом от степени формализации и автоматизации проектирования. Это решается, как уже говорилось выше, с помощью CASE-системы, в которой

формализуемые процедуры проектирования, как правило, описаны в виде соответствующих моделей, алгоритмов и запрограммированы.

6. Необходимо минимизировать исходные данные. Получение исходных данных для проектирования систем типа территориально распределенных БД трудоемкий процесс, связанный, прежде всего, с большими организационными трудностями. Причем достоверность этих данных также не может быть велика из-за динамики развития самой системы, заключающейся в частичном изменении структуры, функций и других параметров системы управления.

7. Синтез оптимальных информационных структур распределенных БД относится в основном к классу задач нелинейного целочисленного программирования. Поиск точного решения в этих задачах сводится к полному либо частичному (направленному) перебору. Причем обычно очень легко найти достаточно эффективный эвристический алгоритм решения этих задач, дающий хорошее приближение к оптимальному решению, в то время как поиск точного решения требует трудоемких, а иногда нереализуемых вычислений.

В целом можно выделить *следующие основные этапы в процессе синтеза оптимальной информационной архитектуры ИО распределенной корпоративной АИС.*

1. *Структурно-функциональный и информационный анализ системы* (построение экономико-организационной модели). На этом этапе проводится анализ целей, задач, функций системы. Определяются основные наборы информации, необходимой для реализации задач и функций системы. Производится оценка объемов информации. Определяются основные потребители и источники информации, проводится анализ направлений и интенсивности основных потоков информации. Этап заканчивается составлением технического задания.

2. *Выбор критерия синтеза оптимальной информационной структуры АИС.* На этом этапе определяются основные проектируемые параметры БД и выявляются главные факторы, влияющие на эти параметры, и формируется критерий оптимальности.

3. *Выбор общей архитектуры распределенной БД.* На этом этапе проводится выбор некоторых параметров, не вошедших в критерий, и отбрасывание вариантов структур, не удовлетворяющих заданным ограничениям и функциональным требованиям. Далее формируются стратегия поиска и обновление информации в распределенной БД.

4. *Синтез оптимальной логической структуры баз данных.* На этом этапе, на основании уточненных исходных данных, согласованных с будущими пользователями системы, производится синтез общей логической структуры баз данных АИС, позволяющей обеспечить

обработку запросов пользователей и оптимальной в смысле выбранного ранее критерия. Далее общая логическая структура условно разбивается на блоки (файлы, фрагменты баз данных, условно выделенные из них) заданного размера с целью дальнейшего оперирования с физически реализуемыми единицами информации.

5. *Разработка физической структуры распределенных БД.* На этом этапе определяются оптимальное число БД, их географическое размещение, размещение данных по БД, размещение данных по иерархии памяти БД, прикрепление пользователей к БД. Оптимизация ведется по выбранному ранее критерию.

1.2.3. Выбор критерия оптимального синтеза распределенной архитектуры ИО АИС

Выбор глобального критерия синтеза оптимальной архитектуры ИО АИС предопределяет локальные критерии создания отдельных БД и влияет на показатели эффективности функционирования системы в целом. Поэтому необходимо, чтобы глобальный критерий был по возможности комплексным, т. е. включал основные показатели функционирования АИС и не противоречил критерию АИС, который по своему содержанию является экономическим.

Рассмотрим некоторые основные критерии применительно к созданию территориально распределенных БД. Сравнительный анализ проводится на основании изучения их влияния на параметры, характеризующие эффективность ИО АИС со следующих точек зрения:

- стоимость хранения информации за определенный период времени;

- стоимость передачи установившихся потоков информации за определенный период времени;

- стоимость обновления баз данных за определенный период времени;

- стоимость поиска данных при установившемся потоке запросов за определенный период времени;

- быстродействие;

- надежность;

- достоверность хранения и передачи информации;

- суммарный объем хранимой информации;

- суммарные объемы передаваемой информации;

- степень равномерности загрузки каналов связи.

Отметим, что среди этих параметров есть зависимые, однако они рассматриваются отдельно из-за их важности.

Рассмотрим некоторые возможные критерии синтеза оптимальной информационной структуры территориально распределенных БД.

МИНИМУМ СУММАРНЫХ ИНФОРМАЦИОННЫХ ПОТОКОВ.

Минимизация потоков информации приводит в основном к уменьшению затрат на передачу информации. Однако этот критерий не учитывает стоимости хранения информации. Очевидно, что при допущении дублирования информации этот критерий не может быть хорошим измерителем эффективности проектируемой системы хранения и обработки информации, так как приводит в пределе к максимальной децентрализации размещения информации, увеличению стоимости хранения и обновления информации. Процесс обновления информации при этом затрудняется из-за наличия большого числа дублирующих данных. Однако децентрализация размещения информации повышает надежность системы. Уменьшается общее быстродействие системы в результате увеличения объемов хранимой информации.

МИНИМАЛЬНЫЙ ОБЪЕМ ХРАНИМОЙ ИНФОРМАЦИИ. С одной стороны, уменьшение объемов хранимой информации уменьшает непериодические затраты на хранение и обновление информации, увеличивает быстродействие поиска информации в одной БД. С другой стороны, уменьшение объемов хранимой информации приводит к увеличению объемов передаваемой информации, общего времени и стоимости поиска информации из-за необходимости обращения к нескольким БД и, тем самым, к увеличению общей стоимости системы. Степень централизации возрастает, что ухудшает надежность системы.

МИНИМАЛЬНАЯ СТОИМОСТЬ ОБНОВЛЕНИЯ ИНФОРМАЦИИ. При наличии дублирующих фрагментов БД и большой их изменчивости потоки обновления информации в территориально распределенной системе становятся весьма значительными, а иногда даже превалируют над остальными потоками. Использование данного критерия при создании распределенного ИО АИС приводит к децентрализации размещения информации по источникам и к сокращению дублирования данных. Следствием этого является уменьшение объемов и стоимости хранимой информации, но увеличение потоков информации, т. е. увеличение стоимости передачи информации. Общее быстродействие системы падает из-за излишней децентрализации информации. Надежность системы повышается.

МАКСИМАЛЬНОЕ БЫСТРОДЕЙСТВИЕ. Это среднее время ответа на запрос пользователя, в которое включаются:

время передачи запроса в “ближайшую” (например, в стоимостном или географическом смысле) БД;

время поиска данных и формирования ответа в СУБД;

время передачи ответа пользователю.

Данный критерий приводит к ужесточению требований к быстродействию ЭВМ, среднему времени доступа запоминающих устройств, скорости передачи данных и т. д. Это удорожает систему, но быстродействие в корпоративных АИС, как правило, не является главным и предопределяющим фактором. Необходимое быстродействие можно обеспечить наложением ограничений на выбор средств и методов передачи, хранения и организации информации в БД.

МАКСИМАЛЬНАЯ НАДЕЖНОСТЬ. Надежность ИО АИС зависит от надежности, прежде всего, технических средств (ЭВМ, каналов связи и т.д.). Увеличение надежности системы приводит к его децентрализации, введению структурной избыточности и увеличению степени дублирования информации. Поэтому повышение степени надежности связано со значительными дополнительными затратами. Заданную надежность можно обеспечить выбором соответствующей общей структуры ИО АИС, определением степени централизации информации и степени дублирования. Однако принимать надежность в качестве главного критерия синтеза нецелесообразно, так как при этом практически ухудшаются все экономические характеристики ИО АИС. Поэтому обычно надежность задается как ограничение при проектировании систем по другим критериям.

Рассмотренные выше критерии влияют существенным образом на основные параметры, характеризующие эффективность ИО АИС. Однако ни один из этих критериев в отдельности не может быть принят в качестве главного (глобального) критерия синтеза оптимальной структуры КАБД, так как они, улучшая отдельные параметры, существенно ухудшают другие, то есть ни один из них не является комплексным.

Наиболее рациональным и измеримым критерием экономического характера является комплексный критерий – *минимум стоимости хранения, обновления и передачи информации за определенный период* (с учетом дублирования информации). Этот критерий учитывает основные рассмотренные выше критерии. В общем случае данный критерий можно представить как функцию C следующих основных параметров:

$$C = F(C_p, C_{xq}, C_{sq}, C_{oq}, L_j, L_{oj}, k_j, V_{oj}, V_{bj}, m, T),$$

где: C_p - стоимость передачи единицы информации;

C_{xq} - стоимость хранения единицы информации в единицу времени на ЭВМ типа q ;

C_{sq} - стоимость поиска единицы информации в БД, реализованной на базовой ЭВМ типа q;

C_{oq} - стоимость обновления единицы информации в БД, реализованной на базовой ЭВМ типа q;

L_j - объем фрагмента, (элемента базы данных $j = 1, \dots, m$);

L_{oj} - средний объем информации j-го фрагмента, (элемента базы данных), обновляемого за одно обращение;

k_j - число хранимых дублей (копий) j-го фрагмента, (элемента базы данных);

m - число фрагментов (элементов базы данных);

V_{oj} - средняя интенсивность потока обновления для j-го фрагмента (элемента базы данных) за время T;

V_{bj} - средняя интенсивность потока использования данных j-го фрагмента (элемента базы данных) за время T;

T - рассматриваемый период времени.

Стоимость передачи единицы информации зависит от следующих основных параметров:

$$C_p = f(r, S, g, V_p, t),$$

где:

S - расстояние между пунктами передачи информации;

г - метод передачи информации;

g - скорость передачи информации;

V_p - объем передаваемой информации;

t - время начала передачи (сеанса).

Будем полагать, что для каждого метода передачи r существует некоторая скорость передачи, оптимальная для данного способа, на которой происходит передача подавляющего объема сообщений.

Таким образом, в первом приближении можно считать, что

$$C_p = f(r, S),$$

и для фиксированного метода передачи г

$$C_{pr} = f(X, Y) = f(S),$$

где X, Y - относительные географические координаты.

Обычно известна арендная (абонентская) плата A_r за использование г-го метода передачи информации в течение времени T. Тогда

$$C_{pr} = A_r / g_{cp} T$$

где g_{cp} - средняя скорость передачи информации за период T .

Далее предполагается, что стоимость передачи информации прямо пропорциональна объемам передаваемой информации.

В общем случае стоимость хранения единицы информации в единицу времени C_{xq} (удельные приведенные к периоду T затраты на хранение единицы информации) является функцией следующих основных параметров:

$$C_{xq} = (aK_q + C_q) / V_x$$

где:

K_q - капитальные вложения на систему накопителей для БД;

q - тип базовой ЭВМ, на которой предполагается реализация БД;

a - коэффициент окупаемости капитальных вложений;

C_q - эксплуатационные расходы за период T на поддержание системы накопителей для БД;

V_x - суммарный объем хранимой информации.

В свою очередь, K_q зависит от затрат на приобретение накопителей, их установку, затрат на строительство и оборудование помещений и других однократных (непериодических) затрат.

Эксплуатационные расходы C_q являются периодическими затратами за время T и зависят от коэффициента амортизационных отчислений, суммы капитальных вложений, абонентской платы за электроэнергию, теплоснабжение, от зарплаты персонала, обслуживающего оборудование и служебные помещения, стоимости расхода носителей информации и других материалов за период T .

Таким образом, C_{xq} включает как непериодические, так и периодические составляющие, которые зависят, в свою очередь, от географических координат. Эта зависимость для различных стран, географических поясов, зон и т. д. очевидна. Менее очевидна она для достаточно малых территорий, таких как, например, город. Однако даже в одном городе стоимость строительства зданий зависит от районов, участка и т. д., так как различаются условия подготовки строительства, подвода коммуникаций. Затраты C_{xq} зависят и от времени, так как со временем меняются цены на оборудование, работы и услуги.

Почти всегда можно оценить C_{xq} , если известна A_q - средняя плата за эксплуатацию ЭВМ типа q с рассматриваемым комплектом накопителей за время T . Тогда C_{xq} можно оценить следующим образом:

$$C_{xq} = (K_n A_q) / V_x$$

где K_n – относительный коэффициент стоимости накопителей к стоимости ЭВМ.

Стоимость поиска единицы информации в БД C_{sq} зависит от среднего времени занятия процессора при поиске единицы информации и от типа базовой ЭВМ q . Стоимость обновления единицы информации в БД C_{oq} зависит в основном от времени занятия процессора при обменах с целью обновления единицы информации.

Таким образом, рассмотренный критерий влияет на основные параметры, характеризующие эффективность БД, а его составляющие достаточно просто измерить. Причем важно, что он учитывает как периодические, так и непериодические затраты.

В заключение отметим, что, несмотря на комплексность выбранного критерия, он не учитывает ряд параметров, характеризующих эффективность БД, таких как быстродействие, надежность, степень типизации и унификации элементов и ряд других. Формирование комплексного критерия, учитывающего все параметры, невозможно и нецелесообразно из-за чрезмерной сложности получаемых при этом моделей синтеза ИО АИС и трудностей реализации алгоритмов их решения на ЭВМ. Поэтому не вошедшие в критерий параметры обычно выбираются на основании логического анализа либо на этапе выбора общей структуры, либо на этапах логической и физической структуризации БД.

При дальнейшей детализации структуры информационного обеспечения АИС разработчики вводят в рассмотрение или уточняют следующие показатели (критерии):

информационная избыточность – дублирование части используемых в АИС данных, которые в наибольшей степени влияют на ее нормальное функционирование и требуют значительного времени восстановления при разрушении;

конфиденциальность информации – обеспечение доступа к засекреченной информации только тому, кому она предназначена;

полнота базы данных - относительное число описаний объектов, хранящихся в базе данных, к общему числу объектов соответствующей предметной области или по отношению к числу объектов в аналогичных БД по той же предметной области;

достоверность базы данных – степень соответствия данных об объектах в БД реальным объектам в соответствующей предметной области в данный момент времени;

идентичность базы данных – относительное число достоверных (не содержащих ошибок) описаний объектов к общему числу описаний объектов в базе данных;

актуальность базы данных – относительное число морально устаревших данных об объектах в базе данных к общему числу накопленных и обрабатываемых данных;

оперативность базы данных – степень соответствия динамики изменения данных при обновлении базы данных состоянию объектов соответствующей предметной области, или величина запаздывания между появлением или изменением характеристик реального объекта и их отображением в базе данных;

форматная совместимость базы данных – степень соответствия данных в БД требованиям стандартов на форматы представления данных для БД соответствующих типов и др.

1.2.4. Выбор параметров архитектуры ИО распределенной АИС

Исходными данными для выбора параметров распределенной архитектуры баз данных корпоративной АИС являются результаты системного анализа системы управления корпорацией, т. е. данные о структуре, задачах, функциях системы, общие характеристики информационных потоков. Предполагается также известным глобальный критерий синтеза распределенных БД. Однако, как было показано выше, не все параметры системы учитываются в этом критерии. При выборе параметров ИО АИС решаются следующие вопросы:

- определение степени централизации информации в БД;

- определение топологии связей между базами данных;

- выбор общей стратегии поиска информации в БД;

- выбор общей стратегии обновления информации в БД.

Определение степени централизации информации в БД. По степени централизации информации можно выделить следующие основные структуры БД:

- абсолютно централизованные;

- абсолютно децентрализованные;

- комбинированные.

Централизация информации в единой системе накопителей БД имеет ряд неоспоримых преимуществ, таких как:

- возможность уменьшения дублирования информации до минимума;

- обеспечение максимальной унификации методов организации, хранения, обновления и поиска информации;

- унификация форматов и структур представления данных;

- возможность унификации СУБД;

- максимальная унификация технических средств.

Однако централизованные структуры наряду с перечисленными преимуществами имеют ряд недостатков, которые ограничивают сферу их применения:

- высокая степень централизации информации увеличивает стоимость эксплуатации в территориально распределенной АИС;
- ухудшается надежность системы в целом.

Общая длина и сложность трактов передачи информации для распределенных БД при централизованном хранении информации может быть значительно больше, чем для децентрализованных, что приводит к потере достоверности информации при передаче данных. Абсолютно централизованные структуры БД находят применение в системах управления малыми городами и отдельными предприятиями, т.е. абсолютно централизованные БД эффективны для локальных АИС.

Децентрализованное ИО АИС – это системы, в которых каждая БД является относительно автономной активной структурной единицей, взаимодействующей с другими БД на равноправных началах. Отсутствие в таких структурах управляющего (центрального) звена приводит к тому, что:

- невозможно избежать дублирования информации и ее неконтролируемого роста;

- существенно увеличивается служебная информация, так как в каждой БД должна храниться полная информация о размещении данных во всех других БД АИС;

- значительно затрудняются процессы размещения и управления данными.

Децентрализованное ИО АИС можно рассматривать как сеть относительно автономных БД, имеющих ограниченную правовыми нормами возможность информационного обмена.

Комбинированная архитектура ИО АИС – это комплекс БД, в котором одна или несколько БД выделены в качестве управляющих {центральных}. Обычно в качестве управляющей выделяется мощная БД, которая кроме выполнения функций управления комплексом БД концентрирует в себе основные общесистемные данные. В комбинированных структурах достигаются частично преимущества централизации информации. Наличие в них управляющих БД позволяет:

- уменьшить служебную информацию за счет централизованного ее хранения в управляющей (центральной) БД;

- уменьшить дублирование информации или контролировать его необходимость по определенным, заранее выработанным критериям и оценкам;

управлять размещением данных в комплексе БД по выработанным заранее алгоритмам и критериям и вести статистический анализ за использованием информации во всем ИО АИС;

разместить информацию, вышедшей из строя БД в другой БД, имеющей резервы памяти.

Комбинированные, территориально распределенные структуры БД получают самое широкое распространение в корпоративных АИС. Структуры таких комплексов не обязательно повторяют организационную структуру управления соответствующих корпораций.

Таким образом, можно сделать следующие выводы:

как правило, в ИО корпоративных АИС должна быть хотя бы одна управляющая (центральная) БД, в которой хранятся общесистемные данные и служебная информация о размещении данных в БД;

в сложных иерархических системах управления может быть и более одной управляющей БД;

число БД на каждом уровне иерархии определяется в основном ограничениями на объемы хранимой информации в одной БД (с точки зрения технических возможностей ЭВМ, обеспечения надежности и живучести БД) и ограничениями на капитальные вложения на создание БД;

основными факторами, определяющими размещение БД, является размещение потребителей (пользователей) и источников соответствующей информации, поэтому главной целью размещения является приближение мест хранения информации к местам ее возникновения и потребления (использования).

Решение вопроса о числе управляющих БД зависит, прежде всего, от следующих обстоятельств:

имеются ли на соответствующем уровне иерархии характерные только для этого уровня данные;

насколько велик объем этих данных (т.е. могут ли быть они структурированы в самостоятельной БД);

существует ли простая возможность получения необходимой информации из других БД и т.д.

Например, при проектировании АИС корпорации может быть принята двухуровневая система ИО. Управляющей БД является центральная БД, в которой должна храниться общекорпоративная информация и структурная (служебная) информация о размещении данных в других БД. Далее по иерархии выделяются БД для предприятий и других структурных единиц корпорации.

Топология связей БД в ИО АИС определяется характером их информационного взаимодействия, а также взаимодействия

пользователей с комплексом БД. Определяющая роль здесь принадлежит структуре связи магистральных каналов. Основными типами структур связей являются (рис. 3а-3е): радиальная, радиально-узловая, кольцевая, каждый с каждым, сеточная, комбинированные.

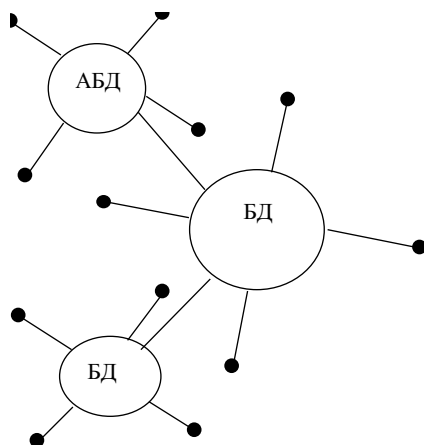


Рис. 3а. Радиальная структура

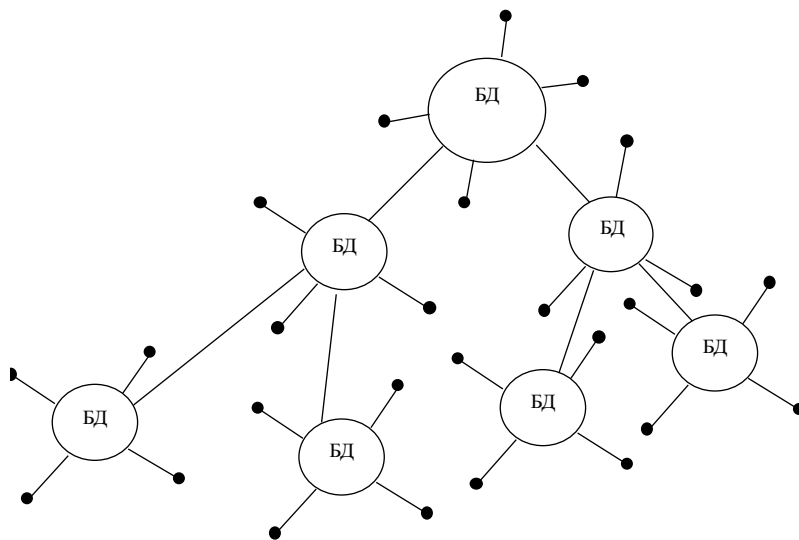


Рис. 3б. Радиально-узловая структура

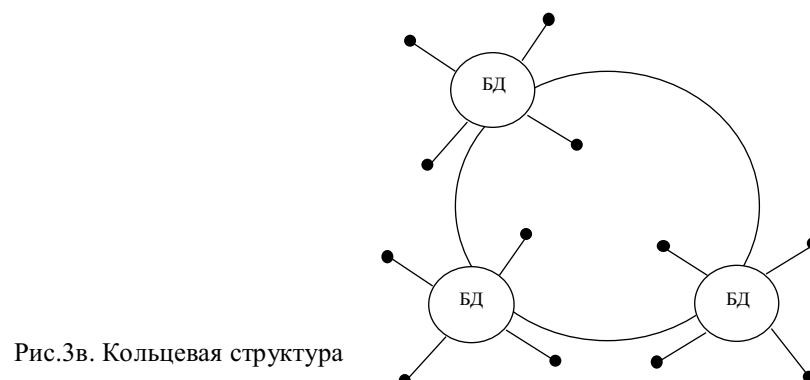


Рис.3в. Кольцевая структура

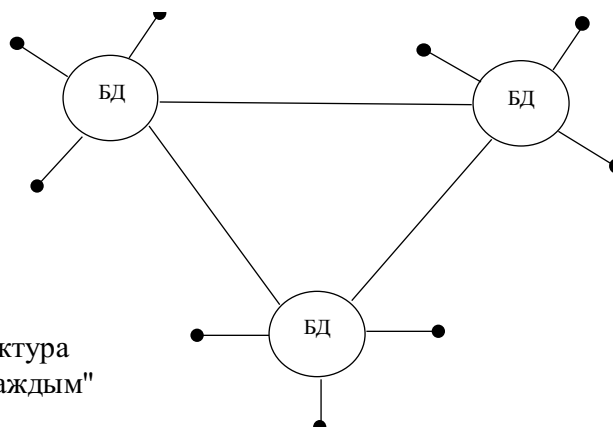


Рис.3г. Структура
"каждый с каждым"

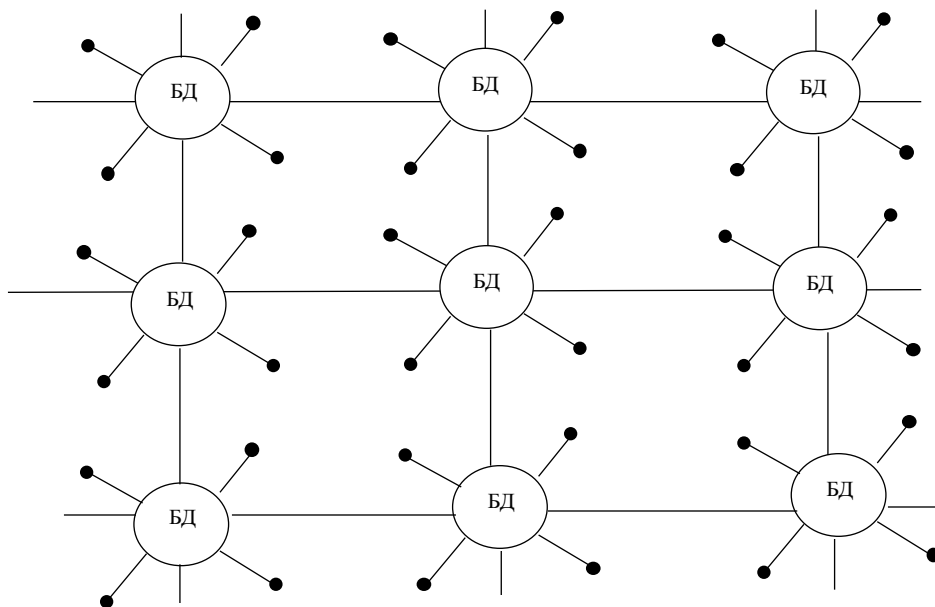


Рис.3д. Сеточная структура

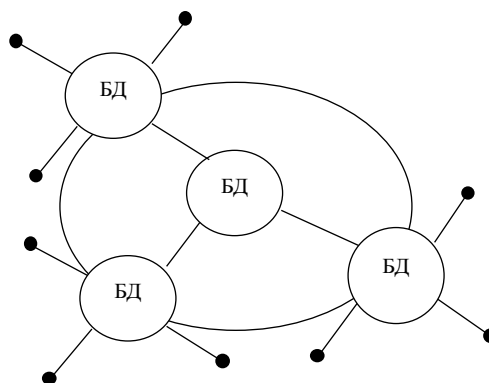


Рис.3е. Радиально-кольцевая структура

Радиальные структуры преимущественно отражают информационные связи непосредственного управления и подчинения. Например, по радиальной схеме могут быть соединены управляющая БД со всеми другими. Радиально-узловая структура является модификацией радиальной, но она требует большего числа каналов связи. Если рассматривать БД как узлы, к которым подключены пользователи, то, соединяя БД с управляющей БД (УБД), получим радиально-узловую структуру. Такая структура обычно отражает иерархию управления и применяется в строго иерархических системах управления.

Радиальные и радиально-узловые структуры отличаются ненадежностью. Более надежные структуры сложнее, дороже и требуют хранения большего количества служебной информации в каждой БД и большего числа каналов связи (каждый с каждым, сеточные, комбинированные). Среди сложных структур наиболее экономичной является сеточная, которая соответствует равномерно распределенной сети равноправных БД, каждая из которых взаимодействует в основном с окружающими (ближайшими в географическом смысле) БД.

Радиально-кольцевая структура соответствует сети БД, где выделена центральная (управляющая) БД, с которой преимущественно взаимодействуют периферийные БД. Кольцевые связи периферийных БД, с одной стороны, отражают их информационные связи, а с другой - обеспечивают надежность ИО АИС. Наиболее надежной является структура каждый с каждым. В качестве преимущества сложных структур подобного типа, кроме упомянутых, следует отметить быстрое действие поиска информации – возможно прямое обращение к любой БД. Информационные связи такого типа возникают обычно среди множества объектов (систем) одного ранга или уровня иерархии, подчиненных друг другу только функционально.

Таким образом, основными факторами, влияющими на выбор определенной архитектуры ИО распределенной корпоративной АИС, являются:

- направления и интенсивность основных потоков информации, которые, в свою очередь, зависят от географического размещения источников, потребителей и мест хранения информации;
- обеспечение заданной надежности, что приводит к необходимости введения структурной избыточности (например, введения дополнительных каналов связи, обеспечивающих функционирование системы при выходе из строя некоторых каналов связи или даже участков сети связи);
- обеспечение заданных достоверности и скорости передачи данных, что осуществляется выбором методов и аппаратуры передачи данных.

1.3. Разработка спецификаций на функции, физический компонент и программное обеспечение АИС

Разработка спецификаций имеет важнейшее значение для последующего правильного функционирования компонентов АИС и системы в целом. **Спецификация в АИС** – это точное и подробное описание действий, которые должна выполнять система, в частности:

спецификация программного обеспечения – это точное и подробное описание действий, которые должна выполнять система программного обеспечения АИС, причем основное внимание уделяется тому, что ПО должно выполнять, а не тому, как это должно выполняться;

программная спецификация – точное описание того результата, который должен быть достигнут с помощью программы, может иметь форму отображения набора входных данных в набор выходных данных;

разработка спецификации требований на ПС – это формализация функций, условий внешней среды, требований к характеристикам и качеству решения задач.

Разработка спецификаций для проектов корпоративных АИС в настоящее время выполняется с помощью систем автоматизированного проектирования, которые позволяют в несколько раз повысить эффективность, скорость и надежность разработки.

CASE-средства поддерживают автоматизированное проектирование спецификаций требований на программы и функциональные группы программ. Повышению эффективности разработки могут значительно способствовать заимствования из предыдущих проектов спецификаций прототипов, версий и отдельных компонентов. Для обеспечения удобства проектирования большое значение имеют графические методы визуализации технических решений и логического контроля проекта.

Созданные спецификации являются базой для детального планирования процесса разработки АИС и ее компонентов. На основе такого плана разрабатывается подробный график работ и выделяются ресурсы для реализации каждого этапа. Этот график в последующем уточняется и корректируется в течение всего времени проектирования. CASE-средства обеспечивают удобства работы с такими графиками, их изменения, выявления критических путей и этапов работ. Руководители проектов должны иметь для управления полную и наглядную информацию:

- о состоянии и развитии всех компонентов проекта;
- об используемых ресурсах;
- об объектах или процессах, имеющих риск нарушения планов.

Структурное проектирование и создание основных спецификаций обеспечивают возможность выбора системы управления базой данных проектирования и распределения ресурсов базы данных проекта.

Для однозначного понимания и описания функций программных и информационных компонентов, функциональных групп программ и структур данных необходима унифицированная система именования и идентификации функций. Описания спецификаций должны содержать иерархически упорядоченную совокупность характеристик компонентов, обеспечивающую:

- их каталогизацию;
- автоматический поиск группы наиболее подходящих из них по их описаниям;
- окончательный подбор компонентов с учетом всей номенклатуры функциональных и технических показателей.

При разработке спецификаций должны выполняться правила структурного построения программных компонентов, контроля функционирования программных компонентов и защиты данных от разрушения.

Информационные модули распределяются по уровням доступности (видимости) из программных компонентов и должны иметь описания, стандартизированные по форме и структуре содержания. Целесообразно использовать унифицированные описания типовых структур данных, выделяемых в информационные повторно используемые компоненты. В спецификациях должна использоваться стандартизированная система правил межмодульных интерфейсов. Типовые операторы взаимодействия компонентов должны обеспечивать гибкость изменения структуры программных средств и БД и устойчивость к ошибкам при таких изменениях.

В спецификациях необходимо предусматривать иерархическую структуру описания, постепенно расширяющуюся от уровня назначения и сокращенной спецификации требований до текстов программ и набора тестов, на которых должен проверяться компонент.

1.4. Разработка и документирование ТП в соответствии с ТЗ и стандартами

Создание и применение АИС обеспечивается документированием этих объектов и процессов для обеспечения интерфейса с пользователями, а также возможности освоения и развития технических и программных средств и баз данных на любых этапах проекта. По своему назначению и ориентации на определенные задачи и группы

пользователей документацию можно разделить на три типа:

технологическая документация, подготавливаемая для специалистов, ведущих разработку, сопровождение и перенос программ и БД на иные платформы, обеспечивающая возможность детального освоения, развития и корректировки ими программ и данных на всем жизненном цикле;

эксплуатационная документация, создаваемая для конечных пользователей АИС и позволяющая им осваивать и квалифицированно применять эти средства для решения конкретных функциональных задач;

исследовательская документация, предназначенная для анализа характеристик, эффективности и качества технологий и объектов проектирования с целью совершенствования методов и средств автоматизации разработки, сопровождения и переноса программ и БД.

Исследовательская документация имеет экспериментальный характер, зависящий от возможных целей исследований. Основная ее задача состоит в фиксировании и обобщении характеристик и процессов разработки и всего жизненного цикла АИС. Этой документацией пользуются в основном руководители, разработчики и исследователи проектов при анализе технологий, планировании новых разработок АИС или их переноса на иные платформы. Из-за разнообразия исследовательских задач этот тип документации практически всегда имеет оригинальный состав и содержание и, как правило, не стандартизируется. Остальные типы документов стандартизируются, прежде всего, для обеспечения высокого качества создания и применения автоматизированных информационных систем. Например, **в состав технического проекта АИС**, как минимум, включаются следующие разделы (документы):

- общесистемная документация;
- функциональная часть;
- информационное обеспечение;
- программно-техническое обеспечение (платформа);
- организационно-правовое обеспечение.

Технологическая документация, непосредственно и в наибольшей степени определяет процессы и эффективность разработки прикладных программ и данных на иные аппаратные и операционные платформы. Стандарты на эту документацию, регламентируют минимальные требования к документам, сопровождающим весь жизненный цикл АИС и БД. В ней отражаются:

стандарты и руководства, регламентирующие процессы разработки и обеспечения качества;

требования к формализации функций, показателям качества АИС и компонентов;

методы и средства достижения показателей качества;

реальные значения достигнутых показателей качества.

Реальные ограничения ресурсов, используемых в процессе разработки, квалификация специалистов, изменения внешней среды и требований заказчика объективно приводят к отклонениям реализации плана разработки, запланированного в ТЗ. Для контроля таких изменений целесообразно предусмотреть и согласовать с заказчиком специальный документ, регламентирующий правила корректировки плана разработки АИС, а также состав и содержание поддерживающей его документации. Необходимое качество объектов формируется и обеспечивается в течение выполнения частных работ каждого этапа жизненного цикла и окончательно удостоверяется документами при его завершении. Для этого рекомендуется формулировать и документировать требования к:

объекту разработки на данном этапе - к его программным и информационным компонентам, а также к интерфейсу между АИС и с внешней средой;

организации, процессу и технологии выполнения совокупности работ каждого этапа;

методам и характеристикам средств автоматизации выполнения работ, обеспечивающим необходимое качество результатов;

методам и средствам контроля, измерения и документирования процессов и результатов выполненных работ.

Выполнение этих требований по существу контролируется путем измерения объектов и процессов разработки. Измерения объектов разработки сводятся к регулярной, поэтапной регистрации характерных для данного объекта показателей качества, а также к сопоставлению их с заданными требованиями. При обнаружении отклонений от требований принимаются меры либо для улучшения реальных показателей, либо по корректировке требований к показателям для данного компонента на контролируемом этапе. Измерения в процессе разработки состоят в контроле запланированного графика работ, принятой технологии и использования ресурсов. Это должно обеспечивать предотвращение ошибок или их, по возможности, раннее обнаружение.

Общие требования к составу и содержанию документов, поддерживающих создание АИС, представлены в ряде стандартов разного ранга, в фирменных описаниях технологий и в публикациях по управлению проектами. В ряде стандартов требования к документации

предписываются как обязательные, а в остальных случаях имеют преимущественно рекомендательный характер. Состав документов широко варьируется в зависимости от класса и характеристик АИС, а также в зависимости от используемой технологии.

Ниже приведен состав основных документов технического проекта АИС в соответствии с государственными стандартами:

технический проект АИС – технический документ (система технических документов), содержащий(щих) детальные технические решения по архитектуре АИС, всем компонентам АИС и точное описание функций АИС;

состав проекта АИС – раздел (документ) технического проекта, в котором указано местонахождение основных проектных решений в соответствующих разделах (документах) ТП АИС;

расчет экономической эффективности АИС – раздел (документ) технического проекта, в котором указаны: методика определения экономической эффективности АИС, расчет затрат на создание АИС, расчет затрат на эксплуатацию АИС, расчет годового экономического эффекта от использования АИС;

ведомость документов ТП АИС – документ, содержащий перечень документов, включаемых в данный ТП, в том числе, заимствованные из других проектов;

общесистемная документация проекта АИС - документация, предназначенная для общего описания и обоснования решений, принятых в проекте, – пояснительная записка, общее описание, расчет экономической эффективности и т.д.;

описание информационного обеспечения (ИО) – раздел (документ) ТП, который содержит описание: принципов организации ИО, классификации и кодирования данных, организации баз данных.

Документы ТП передаются **держателям подлинников технической документации** – организациям, в которых хранятся, на правовой основе, оригиналы всей технической документации по АИС.

В международных стандартах состав проектных документов АИС расширен. Например, стандарт ISO 09007 “Понятия и терминология для концептуальной модели базы данных” вводит понятие “информационной системы” (ИС) и предлагает трехуровневую архитектуру ИС, которая содержит:

- концептуальный уровень;
- внешний уровень;
- внутренний уровень.

Каждый уровень должен быть описан в соответствующей проектной документации.

В этой модели концептуальный уровень соответствует прикладному уровню базовой модели взаимодействия открытых систем (ВОС), а внешний уровень - уровню представления ВОС. Именно на этих уровнях реализуется общение пользователей с базами данных. В соответствии с этими представлениями *концептуальный уровень* ИС состоит из концептуальной схемы, информационной базы и информационного процессора.

Под *концептуальной схемой* понимается описание возможных состояний связей, действующих между объектами в предметной области.

Информационная база - это совокупность описаний объектов в предметной области и текущих состояний их связей.

Информационный процессор - это механизм, нарушающий статичность концептуальной схемы и информационной базы в ответ на внешние команды.

Внешний уровень информационной системы включает внешнюю схему, базу данных, процессор.

Внешняя схема представляет описание связей с точки зрения пользователей и прикладных программ.

Во внешней базе данных информация представляется в соответствии с описаниями внешней схемы.

Внешний процессор отражает связь пользователя и прикладных программ с данными.

На *внутреннем уровне* описываются внутренние и физические представления информации. Информационная база отражает конкретное физическое представление описания предметной области. Внутренний процессор реализует физическое преобразование информации в соответствии с концептуальной схемой. Понятие СУБД в концепции абстрагируется от понятия базы данных и рассматривается как средство для реализации внешнего и внутреннего уровня информационных систем. Подчеркивается, что концептуальная схема, в общем случае, может быть построена вне зависимости от конкретной СУБД.

2. РАБОЧЕЕ ПРОЕКТИРОВАНИЕ АИС И ИНТЕГРАЦИЯ КОМПОНЕНТОВ

2.1. Разработка прикладных программ

2.1.1. Разработка исходных текстов программ

Разработка современных программных средств (ПС) АИС превратилась в мощную программную индустрию, в основе которой лежит понятие **технологического процесса** - части производственного процесса, содержащей действия по изменению и последующему определению состояния предмета производства (программ). Кроме того, используются понятия:

технология программирования - методы, регламентирующие высокий профессиональный уровень написания программ независимо или почти независимо от языка, операционной системы, ЭВМ и решаемой задачи;

обеспечение технологической поддержки и качества ПС - приобретение или разработка и освоение технологии, среды проектирования, средств автоматизации, состава и форм отчетных документов об объектах и процессах разработки.

Структурными единицами программного обеспечения являются:

программный модуль - программа, рассматриваемая как целое в контекстах хранения в наборе данных, трансляции, объединения с другими программными модулями и загрузки в оперативную память для выполнения;

программное изделие - программа на носителе данных, являющаяся продуктом промышленного производства;

программная система (программное средство) - программная продукция, представляющая собой совокупность программ и(или) подсистем, имеющих общее целевое назначение;

программное обеспечение - совокупность программ системы обработки данных и программных документов, необходимых для эксплуатации этих программ;

программное обеспечение общесистемного назначения - программы, обеспечивающие возможность выполнения ЭВМ основных функций, практически не зависящих от специфики конкретных задач и областей применения ЭВМ;

прикладное программное обеспечение - программное обеспечение, предназначенное для решения определенной задачи в предметной области или для предоставления пользователю определенных услуг.

Преобладающей концепцией организации АИС становится *концепция клиент-сервер*, реализуемая большинством фирм-производителей оборудования и операционных систем. При этом клиент-сервер рассматривается как составная часть построения открытых систем. В распределенных, сетевых АИС технология клиент-сервер представляет собой естественное обобщение понятия открытости. Снижение затрат ресурсов достигается созданием специализированных программ - системных менеджеров для планирования рационального распределения ресурсов в вычислительной сети. Концепция клиент-сервер оказалась настолько удачным практическим инструментом создания систем, что вызвала новый виток на рынке операционных систем и оборудования.

Необходимость внедрения архитектуры клиент-сервер поставила целый ряд вопросов в примыкающих областях: проектировании распределенных баз данных и методах проектирования и программирования прикладных систем и общесистемного программного обеспечения. В этой связи принципиально важным является развитие объектно-ориентированных методов проектирования, программирования и построения баз данных, а также основанных на этом подходе интегрированных технологий автоматизированного проектирования информационных систем. Использование архитектуры клиент-сервер дает проектировщикам гибкость и мощь, недоступные ранее, упрощает доступ к данным и другим ресурсам.

Основой программного обеспечения распределенных АИС служат клиент-серверные приложения, которые выполняются в рамках открытых систем. В данном контексте под открытыми системами будем понимать АИС различного назначения, которые размещены в различных узлах телекоммуникационной сети. Работу открытых систем, как единого интегрированного целого, обеспечивает система отраслевых, государственных и международных стандартов в области информационных технологий, специфицирующих интерфейсы, услуги и поддерживающих форматы данных для достижения взаимодействия приложений, данных и персонала. Данные, которые используются в таких приложениях, отличаются сильной информационной неоднородностью. Информационная неоднородность данных заключается в разнообразии:

- предметных областей;
- используемых сущностей, понятий, словарей;
- отображаемых реальных объектов и абстракций в АИС;
- моделируемых процессов;
- видов данных, способов их сбора и обработки; интерфейсов пользователей и т.д.

При разработке и развитии программного обеспечения распределенных АИС объективно необходимо придерживаться соблюдения ряда условий.

1. *Интеграция систем.* Системы эволюционируют от простых, автономных подсистем к более сложным, интегрированным системам, основанным на взаимодействии разнородных компонентов.

2. *Реинженерия систем.* Создание системы и ее реконструкция (реинженерия) - непрерывный процесс формирования, уточнения требований и конструирования. Реконструкция систем осуществляется постепенно. Система должна быть спроектирована так, чтобы произвольные ее составляющие могли быть реконструированы при сохранении целостности системы.

3. *Перенос унаследованных систем.* Любая система после создания противодействует изменениям. Унаследованные системы, часто используют технологии, архитектуры, платформы, программное и информационное обеспечение, при проектировании которых не были предусмотрены нужные меры для их поэтапного переноса в новые системы. Необходимо, чтобы переносимые составляющие системы и оставшиеся компоненты унаследованных систем сохраняли совместимость.

4. *Повторное использование неоднородных информационных ресурсов.* Технология разработки программного обеспечения АИС должна обеспечивать повторное использование информационных ресурсов, переходя от технологии программирования, основанной на интенсивном индивидуальном труде по созданию вручную изделий, удовлетворяющих специфическим требованиям одного конкретного применения, к технологии, основанной на повторно используемых компонентах (ПИК). ПИК должны обеспечивать производство серий стандартизованных продуктов в определенной прикладной области.

Сложность и объемы работ, необходимых для создания эффективных распределенных АИС, требуют объединения усилий различных производителей программного обеспечения. Целью является создание согласованной архитектуры, опирающейся на теорию и практику объектно-ориентированных технологий и общедоступные спецификации интерфейсов информационных ресурсов. Эта архитектура должна обеспечивать повторное использование и мобильность компонентов, опираясь на коммерческие продукты.

Разработка программ на основе объектной технологии вводит концепцию промежуточного слоя. Концепция *промежуточного слоя* - это сосредоточение специальных сервисов в слое архитектуры, расположенном между операционной системой и средствами

управления компьютерными сетями и прикладными системами, специфическими для конкретных АИС. Традиционно к такому промежуточному слою относились средства:

- управления и доступа к данным;
- разработки программ;
- управления распределенными вычислениями;
- поддержки пользовательского интерфейса и др.

Таким образом технология сводится к следующему:

вводится базовая объектная модель унифицированного языка спецификаций интерфейсов объектов;

отделяется реализация приложений от спецификации их интерфейсов;

вводится общий механизм поддержки взаимодействия объектов.

Тем самым достигается однородность представления компонентов и их взаимодействия.

Далее вводится слой унифицированных служб (сервисов), которые используются при проектировании прикладных систем и для формирования функционально законченных средств промежуточного слоя, предлагающих конкретные виды услуг. Службы и средства представляются однородно своими объектными интерфейсами, что позволяет обеспечить их взаимодействие.

Например, функции СУБД представлены рядом таких служб, как:

- долговременное хранение объектов;
- управление конкурентным доступом к объектам;
- служба транзакций;

службы объектных связей, объектных запросов, изменений объектов и т.п.

Эти и другие службы, взятые вместе, реализуют функции как объектных, так и реляционных СУБД. Спецификация служб формируется на основе опыта создания АИС промышленных корпораций.

2.1.2. Прикладное программирование на базе повторно используемых компонентов

Технологии разработки программ являются проблемно-ориентированными, так как каждая из них наиболее эффективна для разработки определенного типа программных средств.

На повышение эффективности процессов создания, эксплуатации и сопровождения прикладных программ АИС направлены методы и средства программной инженерии. Современные концепции построения таких систем основаны на принципах активного исполь-

зования методического, технологического, алгоритмического и программного *задела из предшествующих проектов*. Они предполагают и рекомендуют максимальное повторное использование в новых проектах созданных ранее прикладных ПС и их компонентов, а также переносимость ПС без изменений с одной аппаратно-программной платформы на другую. При разработке и сопровождении сложных прикладных ИС для государственного, регионального, отраслевого управления, для информатизации крупных компаний, коммерческих и финансовых структур используются следующие методы и средства программной инженерии:

- создание переносимых приложений, которые могут работать на основных типах компьютеров и операционных систем и с любыми из распространенных баз данных;

- обеспечение открытости средств разработки, возможности их интеграции с другими средствами и работы в разнородных операционных и сетевых средах;

- проектирование приложений в архитектуре клиент-сервер, позволяющей распределять системные ресурсы в соответствии с требованиями прикладной задачи и оптимизировать использование ресурсов каждого компьютера сети для решения общей задачи;

- использование реляционных баз данных, составляющих основу для накопления, хранения и использования больших объемов разнородных данных для АИС, а также при их создании и сопровождении;

- использование объектно-ориентированного подхода при создании прикладных систем;

- применение возможностей быстрого создания приложений и быстрого прототипирования;

- использование средств управления проектами и коллективной разработки.

В настоящее время почти не производится разработка прикладных систем “с нуля” с целью добиться наибольшей эффективности функционирования программы, поскольку предпочтение отдается быстрому созданию прикладной системы, а недостаточная эффективность функционирования в значительной степени может компенсироваться выбором более мощного оборудования. Напротив, построение прикладной системы из стандартных, хорошо отработанных и переносимых элементов снижает совокупные затраты на создание и сопровождение прикладных систем. Перенос может осуществляться “послойно”, так что за счет переносимости или стандартизации компонентов ПС снизу-вверх можно добиться полной переносимости создаваемых продуктов или делать продукты

переносимыми, реализуя в них только необходимый интерфейс или протокол. Повышение эффективности разработки в целом достигается за счет:

- регламентации порядка проведения работ;
- автоматизации этапов и операций;
- разделения труда между специалистами разной квалификации;
- проблемной ориентации применяемой технологии.

Современный технологический процесс создания сложных программных средств можно разделить на четыре достаточно специфических процесса, каждый из которых имеет особые методы и средства автоматизации:

- системный анализ предметной области, планирование, организация и управление разработкой;
- разработка и накопление программных и информационных компонентов для их многократного применения в определенной проблемно-ориентированной области создания АИС;
- сборка, отладка и испытания базовых версий ПС из подготовленных программных и информационных компонентов;
- модификация и развитие функций версий ПС, а также состава их компонентов для изменения и расширения характеристик базовых версий ПС.

Современная технология проектирования программных средств естественно делится на *технология разработки компонентов* и на *технология сборки ПС из подготовленных компонентов*.

Первая технология должна обеспечивать возможность создания модулей и групп программ в качестве комплектующих изделий с четким описанием функций и взаимодействия с другими компонентами, а также гарантии качества выполнения этих функций, стандартов и соглашений о связях. Для этого компоненты подвергаются автономным испытаниям и проверкам в составе некоторых ПС, а также снабжаются документацией, достаточной для их правильного применения в различных ПС.

Вторая технология использует программные и информационные компоненты как законченные комплектующие изделия и должна обеспечивать их корректную сборку и сопряжение в сложных программных средствах. Эти сложные ПС подлежат аттестации и испытаниям на соответствие заданным требованиям заказчика. В этом случае принципиально изменяются масштабы и сложность объектов, являющихся базовыми в технологическом процессе. Ими становятся программные и информационные модули, в которых *операторы и операнды* скрыты и недоступны для изменения при сборке.

При переходе к сборочному проектированию на базе программных и информационных модулей снижается универсальность использования компонентов. Функциональное содержание модулей специализируется и расширяется. Соответственно вероятность применения каждого из них значительно снижается по сравнению с вероятностью применения операторов языка программирования. Описание функций каждого модуля и условий его сопряжения с другими содержит значительный объем информации, которую трудно представить в полностью формализованном виде. Поэтому для описания приходится применять *языки спецификации* и естественные языки, что приводит к увеличению неопределенностей применения каждого компонента.

Языки спецификации относятся к формализованным информационным языкам, созданным на базе естественных языков путем наложения ограничений на их лексику и грамматику, а также путем применения специальных обозначений для элементов этих языков. Эти языки предназначены для описания и обработки данных.

При рабочем проектировании сложных ПС возможны две стратегии выделения и накопления повторно используемых ПИК.

При *первой стратегии* наборы компонентов выделяются из одного или нескольких комплексов программ, являющихся прототипами для последующей серии программных средств. При этом используются не только методы и алгоритмы решения предшествующих аналогичных задач, но и конкретный опыт реализации программ в функционирующих АИС. Новые ПС базируются на имеющемся программном заделе и их разработка, в значительной степени, ведется по принципу снизу-вверх.

При *второй стратегии* перечень и функции ПИК определяются в процессе системного и структурного проектирования новой проблемно-ориентированной совокупности программных средств. При этом отсутствуют прямые прототипы ПС и основой для проектирования является постановка задачи, функции и технические требования к разрабатываемому ПС, а также системные прогнозы возможного развития аналогичных ПС. Проектирование ПС ведется по принципу сверху-вниз с последовательной детализацией функций компонентов и каждого программного средства.

Технологический процесс разработки ПИК требует накопления, обеспечения использования и модификации версий ПИК. Эти компоненты различаются этапами применения, уровнем отлаженности, характеристиками качества и доступностью для изменений. Кроме того, ПИК приобретают статус завершенных комплектующих изделий с соответствующим документальным оформлением и возможностью

применения в различных сочетаниях и для переноса на иные платформы. Технология разработки ПИК поддерживается средствами автоматизации, которые обеспечивают:

- выделение ПИК, их идентификацию, описание функций и связей;
- разработку ПИК, обеспечение качества и испытания;
- накопление, упорядочение и хранение ПИК;
- поиск необходимых ПИК по их функциям, алгоритмам и описаниям;
- сопровождение и модификацию имеющихся компонентов, а также расширение номенклатуры ПИК, доступных для применения.

Для однозначного понимания и описания функций программных и информационных компонентов, функциональных групп программ и структур данных необходима унифицированная система именования и идентификации функций ПИК. Эта система должна обеспечивать полное и корректное выделение необходимых ПИК из множества подобных версий, различающихся отдельными характеристиками, без необходимости дополнительного анализа конкретных алгоритмов, текстов программ и детальных описаний данных.

При разработке ПИК должны выполняться правила структурного построения программных компонентов, обеспечивающие их потенциальную корректность, контроль функционирования и защиты данных.

В случае, когда разработаны и имеются в базе данных проектирования большинство ПИК, отлаженных в составе базовых версий ПС, основой технологии является их перенос и сборка в новой версии ПС.

Необходимость разработки новых ПИК практически отсутствует и технология сборки реализуется в наиболее полном виде. Особенности новой базовой версии ПС могут вызывать необходимость корректировки некоторых ПИК. Эти изменения оцениваются по степени влияния на все ранее созданные версии ПС, в которых применяются откорректированные ПИК и, если необходимо, переносятся в некоторые из них.

Основой технологии сборочного программирования и переноса на иные платформы является *развивающаяся база данных проектирования* ПС и используемых программных и информационных компонентов, апробированных и пригодных для многократного применения. База данных проектирования по мере необходимости пополняется новыми разработанными ПИК или обновленными версиями старых компонентов, с расширенными или модифицированными функциями. В отдельных разделах базы данных проектирования накапливаются и хранятся базовые версии ПС. Эти

версии оформляются вместе с тестами и результатами испытаний, а также с полной документацией, необходимой для применения, переноса и сопровождения.

Иерархическая модульная структура ПС обеспечивает технологию проектирования по принципу сверху-вниз с позиции учета функционального назначения и наилучшего решения целевой задачи всей АИС. Это способствует концептуальному единству алгоритмов и программ и возможности рационального распределения ограниченных ресурсов проектирования по мере декомпозиции компонентов. Многоуровневое, иерархическое построение ПС позволяет ограничить и локализовать на каждом из уровней соответствующие ему компоненты, вследствие чего облегчается анализ и синтез компонентов более высоких уровней и комплекса в целом.

2.1.3. Трансляция, отладка, тестирование и оценка качества программ и их соответствия требованиям ТЗ

Отладка представляет собой процесс поиска, локализации и устранения ошибок в программах. Наличие ошибки в программе проявляется разными путями:

программа не завершается (зацикливается или ее обработка прерывается до выдачи результата);

программа завершается, но результат выдается неверный;

выдается неверный результат спустя некоторое время после начала работы программы.

В двух первых случаях, когда ошибка явно существует, продолжается отладка. В третьем случае ошибки наиболее неприятные и могут привести к серьезным последствиям, так как обнаруживают себя тогда, когда программа уже сдана в эксплуатацию. Ошибки такого рода возникают в больших програмных системах и имеют место в тех логических ветвях программы, которые не были проверены в процессе отладки и редко активизируются при реализации программы.

Следовательно, отладка является важным этапом в процессе разработки программы, влияющим на успешную реализацию программы. Причем отладка плохо спроектированной программы может занять более 70% времени разработки. Использование структурных методов разработки программ, включающих правильную организацию отладки, значительно ускоряет процесс отладки и облегчает нахождение ошибок.

Условно ошибки можно разделить на синтаксические и логические.

Синтаксические ошибки состоят в нарушении формальных правил написания программы и появляются в результате недостаточного знания

пользователем языка программирования, а также невнимательности при технической подготовке программы к обработке в ЭВМ. К синтаксическим ошибкам можно отнести неправильную запись ключевого слова, отсутствие описания массива, пропуск скобки в арифметическом выражении либо инструкции, задающей формат, и т.д.

Логические ошибки подразделяются на ошибки алгоритма и семантические ошибки.

Ошибки *алгоритма* возникают при несоответствии алгоритма поставленной задаче. Это, прежде всего, ошибки спецификации, неверная запись расчетной формулы, неправильная обработка поискового запроса и т.д.

Ошибки *семантические* (смысловые) являются следствием неправильного понимания программистом смысла инструкций языка программирования или недостаточного знания операционной системы. Например, неправильное обращение к устройствам ввода-вывода, неправильное обращение к процедуре.

Отладка состоит из трех взаимосвязанных действий:

- 1) контроль правильности программы;
- 2) локализация ошибок, обнаруженных в процессе контроля;
- 3) исправление ошибок.

Перечисленные действия могут многократно повторяться.

В условиях современных технологий программирования значительная часть ресурсов, используемых при отладке сложных программ, расходуется на этапе сборки ПС из повторно используемых и закупаемых компонентов.

Технология сборки ПС на базе ПИК должна обеспечивать контроль управляющих и информационных связей между компонентами. Структура интерфейсов определяется правилами структурного построения ПИК. При сборке ПС необходимо установить тождественность конкретных управляющих и информационных взаимодействий в каждой паре вызывающих и вызываемых программных модулей, а также интерфейсов с внешней средой.

Сборка базовой версии ПС завершается комплексной отладкой и испытаниями программ в реальной внешней среде или при ее имитации. Для имитации необходимо создавать модели внешней среды, способные генерировать детерминированные и стохастические тестовые данные, а для соответствующего класса ПС также тесты в реальном времени. Примененные при испытаниях ПС модели внешней среды, исходные данные при генерации базовой совокупности тестов, а также результирующие данные испытаний каталогизируются и накапливаются в базе данных вместе с подготовленной версией ПС. Изменениям в

процессе сопровождения базовых версий ПС должны соответствовать изменениям тестов и результатов испытаний.

Тест - это совокупность специально подобранных входных данных, используемых для контроля правильности работы программы, а *тестирование* - испытание программы на множестве тестовых ситуаций с целью обнаружения ошибок.

Из вышесказанного ясно, что подготовка тестирования включает два шага:

- проектирование тестов;

- подготовка программы для тестирования (вставка дополнительных сообщений, специальных операторов и т. д.).

Проектирование тестов - важный и сложный процесс. Это связано с невозможностью "полного" тестирования программы, т.е. испытания всех режимов работы на всех возможных наборах данных. Стратегия проектирования - уменьшить эту неполноту и с помощью ограниченного набора тестов обеспечить достаточно высокую степень безошибочности работы программы.

Содержание теста определяется спецификацией задачи и логикой ее решения. Поэтому разработка тестов должна предшествовать этапу кодирования алгоритма и осуществляться параллельно с проектированием алгоритма.

Иначе говоря, разработка тестов является предусмотренным заранее и запланированным этапом. К моменту завершения проекта наборы тестовых данных также должны быть спроектированы. Эти наборы являются одновременно обязательным элементом документации программного продукта.

Следуя общим критериям качества, необходимо соблюдать следующие принципы разработки пакета аттестационных тестов для проверки ПС на соответствие ТЗ:

- набор аттестационных тестов должен покрывать весь набор реализуемых функций;

- программы, входящие в пакет, должны быть независимыми в том смысле, что результаты запуска любой из них не должны влиять на условия или результаты запуска любой другой тестовой программы;

- каждая тестовая программа необходима для проверки соответствия между реализацией некоторой языковой конструкции и описанием в спецификации; каждой отдельной спецификации должен соответствовать ровно один тест.

Тесты и тестирование используются для установления соответствия созданных объектов техническим требованиям или заданиям на них, а также для определения достигнутых показателей качества. В

технической диагностике тестом называется последовательность наборов сигналов, которые подаются на вход объекта тестирования и соответствующие им наборы эталонных правильных сигналов, которые должны быть получены на выходе. Для каждого тестового набора указываются координаты (точки) ввода исходных данных и координаты (точки) контроля результатов. Кроме того, для обеспечения корректности тестирования необходимо задавать *допуски на отклонение* реальных результирующих данных от эталонных, в пределах которых следует считать, что полученные результаты соответствуют эталонным. Степень отклонения получаемых результатов от эталонов используется для оценки качества изделий и соответствия их техническим требованиям.

Тестирование у разработчика является органической частью процесса подготовки программ для приемо-сдаточных испытаний у заказчика. Оно доказывает, что обеспечивается необходимая уверенность в том, что должным образом оттестированная программная продукция соответствует требованиям ТЗ, стандарту или другому нормативному документу. Испытания, в свою очередь, определяются как техническая операция, заключающаяся в установлении одной или нескольких характеристик данной продукции, процесса или услуги в соответствии с установленной процедурой.

Основной целью *тестирования для обнаружения ошибок* является выявление отклонений результатов функционирования реальной программы от заданных эталонных значений. Задача состоит в обнаружении максимального числа ошибок, в качестве которых принимается любое отклонение результатов от эталонов. Успешным является тестирование, которое приводит к обнаружению существования ошибок. С этих позиций тесты, не способствующие при испытаниях обнаружению ошибок и только подтверждающие корректность функционирования программ, являются неэффективными.

Цель *тестирования для диагностики и локализации ошибок* - точно установить первичное место искажения программ или данных, являющегося причиной отклонения результатов от эталонных при тестировании для обнаружения ошибок. Эффективными являются тесты, способствующие быстрой и точной локализации первичных ошибок. На этой стадии затраты оправданы и тестирование можно считать успешным, если оно привело к определению элементов программы, подлежащих корректировке.

Тестирование сложных программ имеет свои особенности, которые отличают этот процесс от традиционного, применяемого для проверки аппаратуры и других технических изделий. С этой позиции основными особенностями процесса тестирования программ являются:

отсутствие полностью определенного достоверного эталона (программы), которому должны абсолютно точно соответствовать все результаты тестирования проверяемой программы;

высокая сложность программ и принципиальная невозможность построения тестовых наборов, достаточных для их исчерпывающей проверки;

относительно невысокая степень формализации критериев качества тестирования и достигаемого при этом качества объектов испытаний;

наличие в программах вычислительных и логических компонентов, а также компонентов, характеризующихся стохастическим и динамическим поведением.

Созданы и применяются две группы методов тестирования:

статические - использующие только исходные тексты программ, преимущественно на языках высокого уровня без исполнения той же программы в машинном коде на ЭВМ;

динамические - при которых тестируемая программа исполняется на ЭВМ в соответствующем машинном коде, а тексты в символьном представлении на языке программирования используются как вспомогательные.

Конечной целью тестирования всегда является получение корректной и надежной программы, функционирующей в машинном коде на ЭВМ. Тестирование программы в символьном представлении способствует повышению корректности программы в объектном представлении и наоборот. Процесс тестирования в символьном представлении во многих случаях экономичнее, чем в машинном коде.

Подведем некоторые итоги. Основной целью тестирования является обнаружение, локализация и устранение ошибок в программах. В общем случае под ошибкой подразумевается неправильность, аномалия или неумышленное искажение объекта или процесса. При этом предполагается, что известно правильное, эталонное состояние объекта, по отношению к которому определено наличие отклонения-ошибки. Исходным эталоном для любого ПС являются техническое задание или спецификация требований заказчика (потенциального пользователя), предъявляемые к создаваемым программам. Подобные документы устанавливают состав, содержание и значения результатов, которые должен получать пользователь при определенных условиях и исходных данных. Любое отклонение результатов функционирования программы от предъявляемых к ней требований и сформированных по ним эталонов, следует квалифицировать как *ошибку в программе*.

На этапах трансляции, отладки и тестирования происходит оценка качества разработанных программ и программных комплексов. Качество

любого изделия представляется набором показателей, отражающих его свойства и определяющих возможность и эффективность его применения по прямому назначению. Чем сложнее изделие, тем большее число характеристик описывают его свойства и тем больше необходимо показателей для адекватного отражения его качества.

На практике важно оценивать качество программ не только в завершённом виде, но и в процессе их разработки. Качество объектов достигается, прежде всего, за счёт высокого качества технологических процессов при их создании и управления этими процессами.

Общую проблему обеспечения высокого качества сложных ПС можно разделить на две крупные группы задач:

- создание методов, технологий и средств автоматизации разработки и контроля качества процесса проектирования программ;

- создание методов, методик и средств измерения значений показателей качества программ, разработка которых полностью завершена.

Вторая группа задач призвана обеспечить измерение и регистрацию достигнутых показателей качества программ после завершения процесса их разработки. Их решение позволяет только констатировать имеющиеся характеристики и не влияет непосредственно на их изменения. Для этого применяются проблемно-ориентированные методики тестирования, предназначенные для установления соответствия характеристик готовых к эксплуатации программ технической документации. Такие методики обычно используются группами специалистов, не зависящих от разработчиков, и гарантируют определение реального качества каждого ПС. Если измеренные показатели не соответствуют требованиям технической документации или пользователя, то для их изменения необходимо решать первую группу задач.

Набор показателей качества ПС зависит от функционального назначения и свойств каждого ПС. Достижение необходимых значений показателей на промежуточных этапах имеет генеральную цель - обеспечение в конечном продукте всей номенклатуры показателей качества, заданных в техническом задании. Каждый критерий может использоваться, если определена его метрика и может быть указан способ ее измерения и сопоставления с требуемым значением. В соответствии с принципиальными особенностями ПС выбираются номенклатура и значения показателей качества, которые отражаются в техническом задании, в технических условиях и в спецификации требований на конечный продукт.

Обычно среди показателей качества выделяется две крупные группы и соответствующие им наборы критериев:

1) функциональные критерии отражают специфику областей применения и степень соответствия программ их основному целевому назначению;

2) конструктивные критерии инвариантны к целевому назначению программ и отражают эффективность использования программами ресурсов вычислительных средств, а также надежность и другие общие характеристики функционирования ПС.

Более подробно вопросы обеспечения и оценки качества программ и АИС в целом будут рассмотрены в первой главе юниты 4.

2.1.4. Документирование прикладных программ

Далее перечислены основные документы, которые согласно ГОСТам “Единая система программной документации” должны сопровождать процесс разработки программных средств:

программная документация - документация, содержащая сведения, необходимые для разработки, изготовления, сопровождения и эксплуатации программ;

разработка исходных текстов программ - запись программы на исходном языке программирования с необходимыми комментариями;

описание программы - сведения о логической структуре и функционировании программы;

пояснительная записка к программе - схема алгоритма, общее описание алгоритма и функционирования программы, обоснование принятых технических решений;

описание применения программы - сведения о назначении программы, области и методах применения, классе решаемых задач и ограничениях и о программно-технической платформе;

руководство программиста - сведения, необходимые для установки, настройки и эксплуатации программы;

руководство оператора - программный документ, содержащий сведения для обеспечения процедуры общения оператора с исполнительной системой в процессе выполнения программы.

2.2. Разработка баз данных

2.2.1. Структуры баз данных

Основные идеи современных информационных технологий базируются на концепции, согласно которой данные должны быть

организованы в базы данных (БД) с целью адекватного отображения изменяющегося реального мира и удовлетворения информационных потребностей пользователей. Эти базы данных создаются и функционируют под управлением специальных программных комплексов, называемых системами управления базами данных (СУБД).

Увеличение объема и структурной сложности хранимых данных, расширение круга пользователей информационных систем привели к широкому распространению наиболее удобных и сравнительно простых для понимания реляционных (табличных) СУБД.

В данном курсе будут рассмотрены только структуры реляционных баз данных и системы управления реляционными базами данных, что связано с их наибольшим распространением в современных корпоративных АИС.

Этот подход является наиболее распространенным в настоящее время, хотя наряду с общепризнанными достоинствами обладает и рядом недостатков. К числу наибольших достоинств реляционного подхода можно отнести:

- наличие небольшого набора абстракций, которые позволяют сравнительно просто моделировать большую часть распространенных предметных областей и допускают точные формальные определения, оставаясь интуитивно понятными;

- наличие простого и в то же время мощного математического аппарата, опирающегося главным образом на теорию множеств и математическую логику и обеспечивающего теоретический базис реляционного подхода к организации баз данных;

- возможность манипулирования данными без необходимости знания конкретной физической организации баз данных во внешней памяти.

Реляционные системы далеко не сразу получили широкое распространение. В то время как основные теоретические результаты в этой области были получены еще в 70-х, и тогда же появились первые прототипы реляционных СУБД, долгое время считалось невозможным добиться эффективной реализации таких систем. Однако отмеченные выше преимущества и постепенное накопление методов и алгоритмов организации реляционных баз данных и управления ими привели к тому, что уже в середине 80-х годов реляционные системы практически вытеснили с мирового рынка другие типы СУБД.

Основными недостатками реляционных СУБД являются:

- недостаточная эффективность;

- определенная ограниченность (прямое следствие простоты), присущая этим системам при использовании в таких областях

применения, как, например, системы автоматизации проектирования, в которых требуются предельно сложные структуры данных;

невозможность адекватного отражения семантики предметной области, т.е. возможности представления знаний о семантической специфике предметной области в реляционных системах очень ограничены.

Начнем с рассмотрения на сравнительно неформальном уровне основных понятий реляционных баз данных, а также реляционной модели данных. Необходимо обратить внимание на простоту и возможности интуитивной интерпретации этих понятий. Основными базовыми понятиями реляционных баз данных являются:

- тип данных;
- домен;
- атрибут;
- кортеж;
- первичный ключ;
- отношение.

Рассмотрим смысл этих понятий на примере отношения СОТРУДНИКИ, содержащего некоторую информацию о сотрудниках некоторой организации (рис. 4.)

Понятие *тип данных* в реляционной модели данных полностью соответствует понятию типа данных в языках программирования. В современных реляционных БД допускается хранение следующих типов данных:

- символьных;
- числовых;
- битовых строк;
- специализированных числовых данных (например, “деньги”);
- специальных данных (дата, время, временной интервал).

Кроме того, в некоторых СУБД возможности реляционных систем расширяются введением абстрактных типов данных.

Понятие домена более специфично для баз данных, хотя и имеет некоторые аналогии с подтипами в некоторых языках программирования. В самом общем виде *домен* определяется заданием некоторого базового типа данных, к которому относятся элементы домена, и произвольного логического выражения, применяемого к элементу типа данных. Если вычисление этого логического выражения дает результат “истина”, то элемент данных является элементом домена. Наиболее правильной трактовкой понятия домена является понимание домена как допустимого потенциального множества значений данного типа.

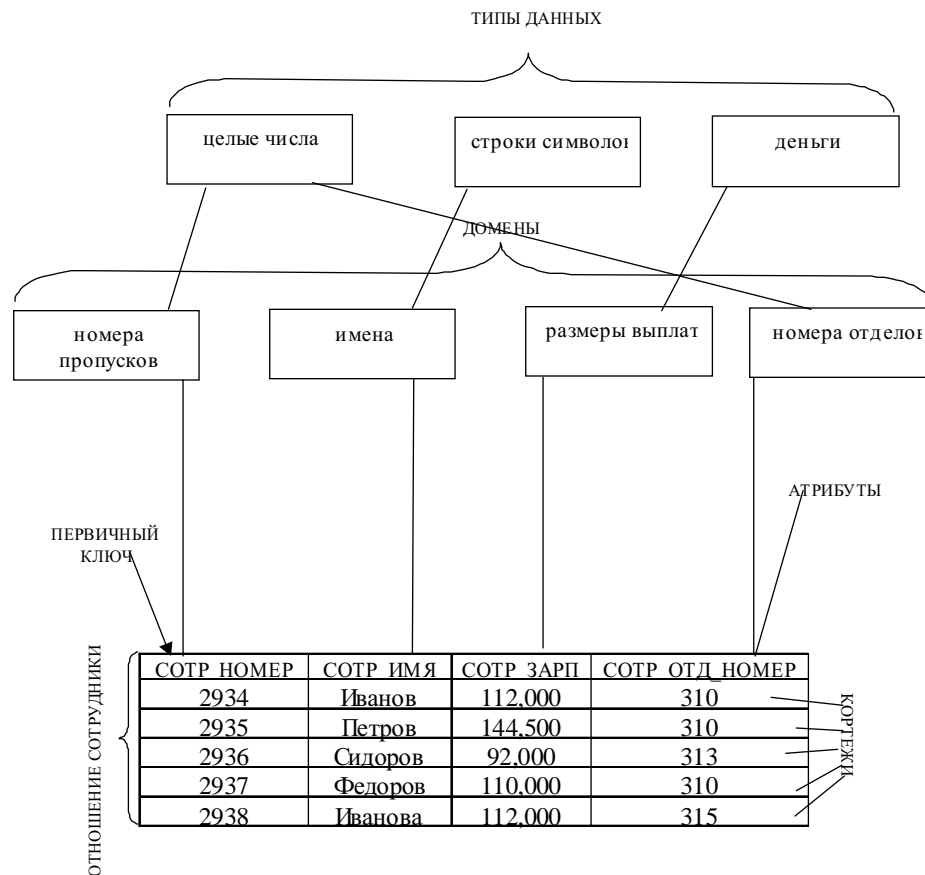


Рис. 4. Иерархия понятий в отношении сотрудники

Например, домен “Имена” в примере определен на базовом типе строк символов, но в число его значений могут входить только те строки, которые могут изображать имя (в частности, такие строки не могут начинаться с мягкого знака). Следует отметить также семантическую нагрузку понятия домена - данные считаются сравнимыми только в том случае, когда они относятся к одному домену. В примере значения доменов “Номера пропусков” и “Номера отделов” относятся к типу целых чисел, но не являются сравнимыми. Заметим, что в большинстве

реляционных СУБД понятие домена не используется, хотя в Oracle V.7 оно поддерживается.

Схема отношения - это именованное множество пар *имя атрибута, имя домена* (или типа, если понятие домена не поддерживается). Степень, или “арность” схемы отношения - мощность этого множества. Степень отношения СОТРУДНИКИ равна четырем, то есть оно является четырехарным. Если все атрибуты одного отношения определены на разных доменах, можно использовать для именования атрибутов имена соответствующих доменов (не забывая, конечно, о том, что это является всего лишь удобным способом именования и не устраняет различия между понятиями домена и атрибута). *Схема базы данных* (в структурном смысле) - это набор именованных схем отношений.

Кортеж, соответствующий данной схеме отношения, - это множество пар *имя атрибута, значение*, которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения. “*Значение*” является допустимым значением домена данного атрибута (или типа данных, если понятие домена не поддерживается). Степень, или “арность” кортежа, т.е. число элементов в нем, совпадает с “арностью” соответствующей схемы отношения. Другими словами, *кортеж* - это набор именованных значений заданного типа.

Отношение - это множество кортежей, соответствующих одной схеме отношения. Некоторые авторы используют термины “отношение-схема” и “отношение-экземпляр”, другие, схему отношения называют *заголовком отношения*, а отношение как набор кортежей - *телом отношения*. Понятие схемы отношения ближе всего к понятию структурного типа данных в языках программирования. В реляционных базах данных, как правило, имя схемы отношения всегда совпадает с именем соответствующего отношения-экземпляра. В классических реляционных базах данных после определения схемы базы данных изменяются только отношения-экземпляры. В них могут появляться новые и удаляться или модифицироваться существующие кортежи. Однако во многих реализациях СУБД допускается и изменение схемы базы данных, определение новых и изменение существующих схем отношения. Это принято называть *эволюцией схемы базы данных*.

Обычным наглядным представлением отношения является таблица, заголовком которой является схема отношения, а строками - кортежи отношения-экземпляра. В этом случае имена атрибутов именуют столбцы этой таблицы. Поэтому иногда говорят “столбец таблицы”, имея в виду “атрибут отношения”. Этой терминологии придерживаются в большинстве коммерческих реляционных СУБД.

Реляционная база данных - это набор отношений, имена которых совпадают с именами схем отношений в схеме БД. Основные структурные понятия реляционной модели данных (если не считать понятия домена) имеют очень простую интуитивную интерпретацию, хотя в теории реляционных БД все они определяются абсолютно формально и точно.

Из приведенных ранее определений следуют ряд важных свойств отношений.

1. *Отношения не содержат кортежей-дубликатов* - это следует из определения отношения как множества кортежей. В классической теории множеств, по определению, каждое множество состоит из различных элементов. Из этого свойства вытекает наличие у каждого отношения так называемого *первичного ключа* - набора атрибутов, значения которых однозначно определяют кортеж отношения. Для каждого отношения, по крайней мере, полный набор его атрибутов обладает этим свойством. Однако при формальном определении первичного ключа требуется обеспечение его "минимальности", т.е. в набор атрибутов первичного ключа не должны входить такие атрибуты, которые можно отбросить без ущерба для основного свойства - однозначно определять кортеж. Понятие первичного ключа является исключительно важным в связи с понятием целостности баз данных.

2. *Отсутствие упорядоченности кортежей отношения* также является следствием определения отношения-экземпляра как множества кортежей. Отсутствие требования к поддержанию порядка на множестве кортежей отношения дает дополнительную гибкость СУБД при хранении баз данных во внешней памяти и при выполнении запросов к базе данных. Это не противоречит тому, что при формулировании запроса к БД можно потребовать сортировки результирующей таблицы в соответствии со значениями некоторых столбцов.

3. *Отсутствие упорядоченности атрибутов отношений* следует из того, что по определению схема отношения есть множество пар *имя атрибута, имя домена*. Для ссылки на значение атрибута в кортеже отношения всегда используется имя атрибута. Это свойство теоретически позволяет, например, модифицировать схемы существующих отношений не только путем добавления новых атрибутов, но и путем удаления существующих атрибутов.

4. *Значения всех атрибутов являются атомарными*, что следует из определения домена как потенциального множества значений простого типа данных, т.е. среди значений домена не могут содержаться *множества значений* (отношения). Принято, что в реляционных БД допускаются только нормализованные отношения (отношения, представленные в 1-й нормальной форме).

Пример ненормализованного отношения показан на рис. 5. Здесь имеет место бинарное отношение атрибута ОТДЕЛЫ, значениями которого являются отношения. Заметим, что исходное отношение СОТРУДНИКИ является нормализованным вариантом отношения ОТДЕЛЫ (см. рис. 6). Нормализованные отношения составляют основу классического реляционного подхода к организации баз данных. Они обладают некоторыми ограничениями (не любую информацию удобно представлять в виде плоских таблиц), но существенно упрощают манипулирование данными.

| номер отдела | сотр номер | сотр имя | сотр зарп |
|--------------|------------|----------|-----------|
| 310 | 2934 | Иванов | 112,000 |
| | 2935 | Петров | 112,500 |
| | | | |
| 313 | 2937 | Федоров | 110,000 |
| 315 | 2938 | Иванова | 112,000 |

Рис. 5. Ненормализованное отношение ОТДЕЛЫ

| сотр номер | сотр имя | сотр зарп | сотр_отд номер |
|------------|----------|-----------|----------------|
| 2934 | Иванов | 112,000 | 310 |
| 2935 | Петров | 144,500 | 310 |
| 2936 | Сидоров | 92,000 | 313 |
| 2937 | Федоров | 110,000 | 310 |
| 2938 | Иванова | 112,000 | 315 |

Рис.6. Нормализованное отношение СОТРУДНИКИ

Рассмотрим, например, два идентичных оператора занесения кортежа:

Зачислить сотрудника X (пропуск номер 3000, зарплата 5000) в отдел номер 320 и зачислить сотрудника X (пропуск номер 3000, зарплата 5000) в отдел номер 310. Если информация о сотрудниках представлена в виде отношения СОТРУДНИКИ, оба оператора будут выполняться одинаково (вставить кортеж в отношение СОТРУДНИКИ). Если же работать с ненормализованным отношением ОТДЕЛЫ, то первый оператор выразится в занесение кортежа, а второй - в добавление информации о X в множественное значение атрибута ОТДЕЛ кортежа с первичным ключом 310.

2.2.2. Ввод информации для контрольных примеров и испытания баз данных

Этот этап включает процедуры испытаний и выбора СУБД, которые состоят из предварительных и окончательных.

Для предварительного выбора предлагаются следующие процедуры. Главный специалист по программному обеспечению совместно с группой специалистов по ЭВМ определяет потенциальные ПС, которые будут затем тестироваться по документальным источникам, описывающим их возможности по управлению данными. При отборе СУБД учитываются:

- высокая рекламная и рыночная наблюдаемость СУБД;
- оценка пользовательской базы (количество уже установленных СУБД);
- авторитет фирмы-производителя;
- функциональные возможности программного средства;
- комментарии покупателей и пользователей.

Когда процедура предварительного отбора закончена, главный специалист заполняет специальные бланки сравнительных характеристик СУБД и передает их на процедуры окончательных испытаний и выбора. Если продукт рекомендован на второй этап (окончательный выбор), главный специалист тщательно просматривает эти пакеты и выбирает СУБД для тестирования, основываясь на следующих положениях:

- насколько хорошо организована документация (без хорошей документации мощные функциональные возможности программ не могут быть никогда использованы полностью);
- соответствуют ли возможности СУБД текущим тенденциям рынка;
- как обеспечен дружественный интерфейс и общая простота использования (СУБД должны быть не только быстрыми и точными, но и очень простыми по языковым возможностям).

Актуальное тестирование выполняется посредством создания стандартных приложений баз данных с использованием одинаковых файлов данных и одинаковых шагов для каждого пакета. Тестирование охватывает все наиболее важные аспекты управления данными. Затем проводится детализация системы значений и весов, определяющих рейтинги показателей качества программ. Это затрагивает те возможности и функции программ управления данными, которые позволяют проводить их сравнение. Значение каждой функции определяется промышленными экспертами в соответствии с ее относительной важностью с точки зрения показателей качества программ в конкретной

области. После выполнения всех шагов первого этапа тестирования главный специалист сравнивает программы пункт за пунктом, их возможности и функции в соответствии с эталоном.

Баллы, полученные в процессе тестирования, определяют общий рейтинг СУБД. Система определения рейтинга включает три уровня, каждый из которых имеет свой вес важности по отношению к следующему более высокому уровню. Система весов для категории теста по СУБД базируется на шкале от 1 (необязательно) до 10 (совершенно необходимо). Разработана система весов для тестируемых СУБД или программ управления файлами. Все рейтинги суммируются и печатаются в карте результирующего рейтинга в отчете по каждому виду СУБД.

После инсталляции программного средства главный специалист изучает его руководство (описание) как на экране, так и в документированном виде. Он обращает особое внимание на глубину и широту предметного описания. Баллы назначаются в соответствии с критериями, и эти баллы затем используются для определения общей рейтинговой оценки программной документации. При этом учитываются:

- время, необходимое для изучения руководства;

- ясность и организация учебника;

- расширение тем - предлагает ли учебник файлы и задачи, стимулирующие пользователя экспериментировать.

Каждая СУБД имеет свои собственные уникальные процедуры доступа, входа, манипулирования данными, свой пользовательский интерфейс. Главный специалист знакомится с базисной логикой СУБД. Такие базисные функции, как структура базы данных и поиск данных, изучаются и тестируются. Однако перед тестированием базисных функций, главный специалист создает тестовую базу данных из нескольких десятков стандартных записей. Эта маленькая база данных затем используется для тестирования всех базисных функций СУБД. При этом учитываются размер записи, размер поля, размер индекса и другие ограничения. Затем создается формат управления входом в одинаковые файлы данных, который используется для ввода тестовых, стандартных записей в базу данных. Программа оценивается по шагам, необходимым для создания новой записи, запоминания записи, и продолжения с входом в следующую запись. Баллы назначаются в соответствии со следующими показателями:

- допустимые типы полей;

- максимальная длина текстового поля;

- допустимые размеры записей и характеристик в файле;

- метод определения структуры файла;

- гибкость в описании формы в расположении полей;

возможности многофайлового просмотра и редактирования полей;
реакции на ошибочные действия и некорректные данные;
простота использования.

Главный специалист формирует три запроса ко вновь созданным файлам данных. Каждая поисковая операция тестирует различные поисковые возможности программы. Первый запрос включает в себя простейшую выборку с проверкой условия “больше чем” или “равно”, два другие включают выборку с использованием операторов “и”, “нет”, “или”. Оценка метода исполнения запросов программой выполняется по отношению к ее производительности с учетом следующих дополнительных критериев:

возможности сортировки и индексации;
генерация отчетов.

Главный специалист использует программу, которая формирует и записывает отчет, если возможно, создает табличный отчет. Записи группируются вместе, для специальных числовых полей среднее определяется по каждой группе, и общий итог печатается для выходного отчета.

Первоначальное знакомство с базисными операциями программ СУБД начинается с анализа общего интерфейса пользователя. Оценки выполняются в целом по структуре команд и/или меню в соответствии со следующими критериями:

какие возможности программы доступны из меню;
какие возможности программы доступны через команды;
простота использования;

насколько ясно представляются на экране предупреждения и сообщения об ошибках.

В процессе тестирования отмечаются все недостатки интерактивной документации и проводится обзор различных компонентов документации, возможности подсказок и качество отпечатанной документации. Рейтинг, заранее определенный для программного руководства, комбинируется с рейтингом для подсказок и обобщенного рейтинга программной документации. Возможности подсказок (помощи) интенсивно используются, когда это необходимо, в течение всего процесса тестирования. Оценки делаются как в положительных, так и в отрицательных опытах использования экранных подсказок относительно достаточности ответов по вопросам использования программных операций. Главный специалист проводит также специальный обзор возможностей подсказок (help) по их структуре и содержанию.

На следующем этапе главный специалист начинает исследовать полную мощность программных средств СУБД. Некоторые из этих тестов

включают стандартные файлы базы данных, которые специально созданы для тестирования сложных преобразований базы данных в реальных рабочих условиях управления файлами. Поскольку они не предназначены для использования в условиях жестких требований, то первоначально не тестируются по категориям безопасности, реляционных возможностей и языка программирования. По мнению специалистов, все СУБД должны быть способны поддерживать различные форматы, для чего используются тестовые файлы:

доступный в нелимитированном стандарте ASCII (называемом также CSV);

табличном ASCII (PRN) или dBase - формате (DBF).

Оценки выполняются по шагам, необходимым для реализации передачи данных.

Главный специалист тестирует возможности защиты данных от несанкционированного доступа установлением следующих пользовательских ключей:

пользователь 1 не имеет доступа к полям, содержащим финансовые данные;

пользователь 2 может добавлять и исключать записи, но не может изменять их структуру;

пользователь 3 обладает полным доступом к базе данных, включая структуру, ключи и т.д.

Затем делается считывание файла из внешней программы. Главный специалист назначает баллы в соответствии со следующими основными критериями:

уровни защиты компонентов, содержащихся в БД;

ориентация (насколько ключи ассоциируются со специальными пользователями);

шифрование файлов данных и их структуры.

Для испытания реляционных возможностей маленький файл, созданный на более ранней стадии теста, присоединяется к большему, тестируемому файлу. Главный специалист создает представление комбинации полей из двух файлов и приводит оценку базисных функций программы.

Затем испытываются средства программирования, включая язык, для чего создается как короткое приложение на базе меню, так и быстрая программа, чтобы сравнить два различных способа и определить затраченное время.

Поскольку рынок программных средств очень насыщен, кроме технических аспектов большое значение для программных продуктов приобретают сервисные услуги, предлагаемые поставщиками, и

гарантии, обеспечиваемые программе. Проверяется политика обновления версий программного продукта, проводимая поставщиком. Предыдущие версии продукта по возможности сравниваются с новой и производится их оценка. Оценивается политика по обучению пользователей, проводимая поставщиком. Опыт поставщика оценивается как часть комплексного анализа программы.

Пользователи, независимо от того, каков уровень их опыта, хотят, чтобы программное обеспечение было легким в использовании. Общий рейтинг продукта на простоту использования складывается из нескольких составляющих. Простота интерактивных процедур определяет возможности программы работать с другими программами над общими задачами.

Показатель *стоимость/производительность* определяет для любой программы, стоит она своих денег или нет. Этот рейтинг балансирует между производительностью программы и ее ценой. Оценка производительности программы есть наиболее общая и важная в данном тестировании. Рейтинг производительности есть среднее от рейтингов программы по разделам базисных функций и сложных функций. Рейтинг для стоимости определяется на основании цен на программы-конкуренты.

Когда тестирование завершено, составляется отчет, обобщающий результаты тестирования. Отчет выполняется на специальных форматах, которые включают:

- информацию по деталям всех разделов теста;
- общее заключение о программе;
- обзор конкурентоспособности программы на рынке;
- информацию о цене продукта и его поддержке;
- обсуждение данных поставщика.

Сравнительные таблицы помещаются в отчете по каждому продукту. Ключевые функции и технические характеристики программ СУБД сопоставляются с их главными конкурентами.

2.3. Интеграция компонентов АИС

2.3.1. Интеграция прикладных программ в функциональные блоки

Набор программных и информационных компонентов, разработанных собственными силами, а также приобретенных на рынке программных средств (ПС), на заключительном этапе рабочего проектирования интегрируется (собирается) в функциональные блоки.

В целом, для распределенных корпоративных АИС, проблема сводится к унификации интерфейсов операционных систем с различными прикладными программами, а также с окружающей средой:

- пользователями;
- базами данных;
- средствами коммуникации.

При этом необходимо решать следующие задачи:

- содействовать облегчению переноса кода прикладных программ на различные платформы;
- способствовать определению и унификации интерфейсов заранее, а не в процессе их реализации;
- учитывать созданные ранее и используемые прикладные программы;
- определять необходимый минимум интерфейсов, для ускорения создания и утверждения документов;
- обеспечивать распределенную обработку данных и защиту информации.

Процедуры интеграции компонентов программного обеспечения регламентируются рядом международных стандартов, которые отражают принципы построения интерфейсов прикладных программ с платформой - операционной системой, через которую осуществляется взаимодействие с компонентами внешнего окружения. Прикладные программы непосредственно не взаимодействуют с внешним окружением, а связаны с ним только через операционную систему. Таким образом, определяющими являются два интерфейса между тремя базовыми компонентами:

- между прикладными программами и платформой - операционной системой (API);
- между платформой и внешним окружением (EEI).

Определяются также общие функции (услуги платформы для этих взаимодействий). Внешнее окружение включает компоненты:

- человеко-машинного взаимодействия с пользователями;
- информационного взаимодействия с внешними устройствами ЭВМ;
- обеспечения сетевых коммуникаций.

2.3.2. Интеграция баз данных, технических и программных компонентов

Различные компоненты АИС могут разрабатываться разными коллективами специалистов, а иногда и разными организациями, но в конечном счете наступает момент, когда необходимо все компоненты

интегрировать в единую автоматизированную информационную систему. Все пользователи АИС взаимодействуют с ней с автоматизированных рабочих мест (АРМ) посредством соответствующих интерфейсов.

Для интеграции АРМов пользователей, операционных систем, прикладных программ и баз данных и обеспечения возможности их развития необходимо выделение и унификация интерфейсных компонентов. Эти интерфейсные компоненты должны согласовывать методы доступа разнородных средств распределенных АИС, отличающихся между собой и использующих различные принципы отображения информации. При этом целесообразно сохранение интерфейсов пользователей с информационными системами при любых изменениях платформ. Для этого разработчики должны предусматривать унификацию архитектуры и основных функций взаимодействия пользовательского интерфейса с приложениями.

На международном уровне упорядочена система непосредственного графического взаимодействия пользователей с базовыми средствами манипулирования графическими образами, которые стандартизированы. Регламентация оформлена *группой стандартов виртуальных терминалов (ВТ) и графики*, которые определяют способы взаимодействия прикладных интерактивных систем, в терминах передачи и манипулирования абстрагированными графическими образами. Услуги базового класса ВТ позволяют:

- устанавливать соединение между двумя пользователями;
- запрашивать и согласовывать подмножества услуг и сервисных параметров;
- передавать и манипулировать структурированными данными, независимо от способа внутреннего представления их информации, используемого каждым пользователем ВТ;
- управлять целостностью взаимодействия с ВТ;
- завершать взаимодействие пользователей согласованно и односторонне.

Организация взаимодействия пользователей с данными в АИС состоит в упорядочении и регламентировании организации и структуры данных, а также правил манипулирования ими: накопления; хранения; изменения; транспортировки.

Стремление эффективно использовать ресурсы ЭВМ при различном объеме, содержании и взаимосвязи данных привели к разработке различных методов их структурирования и организации. Соответственно с организацией данных изменялись методы, средства и интерфейсы манипулирования ими со стороны пользователей, обеспечения их целостности, сохранности и защиты. В результате для управления и

использования относительно небольших объемов слабо структурированных данных развиты *методы и стандарты файловых систем*, а для манипулирования сложными, взаимосвязанными данными развиваются *методы и стандарты управления базами данных*. Между данными, характеризующими объекты в АИС, могут существовать связи, имеющие различный содержательный смысл и степень сложности. Для эффективной работы с данными разработчики учитывают эти особенности и используют соответствующие типы систем управления базами данных (СУБД) - сетевые, иерархические и реляционные.

В результате, на заключительном этапе проектирования АИС, возникает проблема совместимости баз данных для пользователей на прикладном уровне. Многие разработчики, для достижения ясности и дружелюбности интерфейса пользователя, отдают предпочтение реляционной модели данных перед сетевой или иерархической, а также создают объектно-ориентированные интерфейсы конечного пользователя с использованием техники меню и окон.

Наряду с ориентацией на прикладного пользователя, большинство СУБД учитывает и потребности пользователя-программиста, предоставляя ему встроенный язык программирования. Используя этот язык, программист может реализовать нетривиальные приложения по обработке данных, ориентированные на конкретные потребности.

Напомним, что при создании или выборе сетевых СУБД в общем случае требуется найти способ организации баз данных, который бы обеспечивал:

- высокую надежность хранения данных;
- необходимую эффективность при доступе к данным из любого узла сети;
- необходимую целостность баз данных при их совместном обновлении из разных узлов.

Подходы к решению этой задачи различны для территориальных и локальных СУБД. Эти различия объясняются в первую очередь разницей в пропускных способностях каналов связи к данным. Высокая эффективность доступа к данным не может быть обеспечена в территориальных сетях с их трудно предсказуемыми задержками при передаче данных, если не предпринять специальные меры для обеспечения копий данных в нескольких (или всех) узлах размещения БД.

Все вышеперечисленные аспекты учитываются разработчиками при интеграции баз данных с другими компонентами АИС. Такими компонентами, в частности, являются файловые системы.

Интеграция совместимых распределенных систем работы с файлами предусматривается стандартами FTAM (File, Transfer, Access

and Management). Они обеспечивают унификацию методов взаимодействия с файлами на основе модели виртуального файлового хранилища (ВФ). Поддерживаются возможности:

- организации неделимых транзакций взаимодействия с данными;
- усложнения структур используемых данных;
- применения механизма восстановления при сбоях и искажениях данных.

Модель регламентирует согласование различий в файловых хранилищах реальных систем, не изменяя каждого из них. Для идентификации файлов в модели используется составной адрес, содержащий имя файлового хранилища и имя в файловом хранилище. Файл, кроме имени, характеризуется атрибутами общих свойств файла, атрибутами, определяющими его логическую структуру, и дополнительными данными о содержании файла.

Для доступа к элементам данных используются атрибуты, определяющие логическую структуру файла. Определен набор операций, позволяющих изменять состояние файлового хранилища и манипулировать элементами данных с учетом структуры доступа. Для этого используются примитивы сервиса FTAM над целыми файлами и над их элементами.

От набора независимых, совместно используемых файлов базы данных отличает свойство их *целостности*, т.е. глубокой взаимосвязи данных, в которой наибольшее значение имеет функциональная зависимость. Особенно полно механизм контроля целостности формализован на основе произвольных логических утверждений в языке SQL, который служит для удобного и понятного пользователям формулирования обращений к реляционным БД и манипулирования данными. Этот язык является мощным интегрирующим средством в распределенных АИС, так как его стандарт используется в большинстве реляционных СУБД.

Язык SQL, в основном, является языком запросов и манипулирования данными, однако его функции и возможности значительно шире. Он обеспечивает построение эффективных диалоговых систем обработки транзакций со стандартизированными наборами запросов. В языке имеются средства:

- определения и манипулирования схемой БД;
- определения ограничений и контроля целостности;
- определения и уничтожения структур физического уровня для эффективной реализации транзакций;
- авторизации и защиты доступа к данным;
- использования точек сохранения транзакций и откатов при сбоях.

Он включает средства динамической компиляции запросов, на базе которых возможна диалоговая их обработка. Это обеспечивается включением операторов, позволяющих в процессе выполнения транзакций откомпилировать и выполнить любой оператор этого языка. Имеются специальные операторы, поддерживающие встраивание операторов SQL в традиционные языки программирования и, прежде всего, в тексты программ на языке Си. При этом осуществляется последующая компиляция операторов SQL в текст программы на языке Си.

В языке SQL обеспечена защита доступа к данным средствами самого языка. С каждой транзакцией неявно связывается идентификатор пользователя, от имени которого она выполняется. После создания нового отношения все привилегии, связанные с этим отношением, принадлежат только пользователю - создателю отношения. Проверка полномочности доступа к данным происходит на основе информации о полномочиях, существующих во время компиляции соответствующего оператора SQL. Язык SQL, как уже говорилось, реализован в большинстве популярных коммерческих СУБД.

В стандартах, регламентирующих создание распределенных информационных приложений, определен встроенный синтаксис для включения операторов языка манипулирования данными в стандартные прикладные программы на Си, Коболе, Паскаль и т.п. Встроенный синтаксис является сокращенной записью, в которой встроенные операторы SQL заменены в явной форме вызовами процедур БД, содержащими оператор SQL. Стандарт применяется для реализаций в среде, которая может включать:

- языки прикладных программ;
- языки запросов конечных пользователей;
- системы генерирования сообщений;
- системы библиотек программ;
- системы распределенной связи;
- различные средства для создания баз данных и организации обработки данных.

Таким образом, описание SQL является фактическим руководством, которое регламентирует и позволяет интегрировать действия пользователей различных уровней:

- пользователей, создающих базу данных и программистов;
- конечных пользователей информации из базы данных.

Кроме интерфейса с пользователями SQL может играть роль интерфейса между различными СУБД. В этих случаях, в СУБД встраиваются средства формирования запросов к другим СУБД на языке SQL.

2.4. Комплексная отладка и тестирование, разработка плана и методик приемо-сдаточных испытаний АИС

Заключительные комплексные испытания АИС проводятся после интеграции компонентов в реальной или имитируемой среде. Программа испытаний, методики их проведения и оценки результатов выполняются разработчиками на основании утвержденного заказчиком технического задания и спецификаций на компоненты АИС.

Основные задачи заключительных комплексных испытаний АИС следующие:

- корректная проверка заданных характеристик АИС;
- подтверждение пригодности испытываемой АИС к эксплуатации в условиях, определенных технической документацией;
- подготовка системы и документации к приемо-сдаточным испытаниям.

Результаты испытаний фиксируются в протоколах, которые обычно содержат следующие разделы:

- назначение тестирования и раздел требований технического задания и/или спецификации, по которым проводились испытания;
- условия проведения тестирования и характеристики исходных данных;
- результаты испытаний с оценкой их на соответствие требованиям технического задания;
- выводы о результатах испытаний и соответствии определенному разделу требований технического задания.

Протоколы по всей программе обобщаются в отчете о проведении испытаний. После чего делается заключение о соответствии АИС требованиям заказчика и принимается решение о подготовке к приемо-сдаточным испытаниям.

В основном, комплексные испытания заключаются в обработке потоков различных данных, которые должны циркулировать в АИС и в последующей фиксации и анализе результатов обработки.

Например, данные с автоматизированных рабочих мест пользователей отражают реальные характеристики воздействий на АИС с учетом особенностей и квалификации человека. В результате через человека и его характеристики замыкается контур обратной связи управления объектами внешней среды. Такое же замыкание контура автоматизированного управления возможно в аналогах и имитаторах реальных объектов.

Данные натурных экспериментов с объектами внешней среды могут подготавливаться заранее, например, при проверке аппаратной части

информационной системы. Эти данные отражают характеристики и динамику функционирования объектов, которые трудно или опасно подключать для непосредственного взаимодействия с непроверенными программами. Кроме того, такие данные могут использоваться для аттестации имитаторов некоторых объектов внешней среды. Они бывают полезны в тех случаях, когда создание определенных условий функционирования объектов внешней среды очень дорого или опасно и может быть выполнено только в исключительных случаях.

При тестировании необходимо иметь эталонные характеристики данных, поступающих на испытываемую АИС. При работе с реальными объектами зачастую приходится создавать специальные измерительные комплексы, которые определяют, регистрируют и подготавливают к обработке все необходимые характеристики в процессе реального функционирования этих объектов. Такие измерения проводятся при автономном функционировании объектов и при их взаимодействии с испытываемыми компонентами АИС в реальном времени. Результаты измерений используются для определения характеристик качества АИС при работе с реальными объектами.

Синхронизация и обобщение тестовых данных предназначены для упорядочения всех тестов в соответствии с реальным временем их поступления в АИС и для распределения по каналам связи между моделирующей и основной ЭВМ. В результате формируются потоки тестовых данных каждого реального объекта внешней среды, которые вводятся в реализующую ЭВМ в соответствии с логикой функционирования информационной системы через соответствующие каналы устройства сопряжения с моделирующей ЭВМ.

При затруднениях реализации реального масштаба времени при имеющейся производительности имитирующей ЭВМ выделяются модели объектов, на которые отсутствуют воздействия обратной связи от испытываемого компонента. Для таких объектов возможно основную часть имитации производить предварительно вне реального времени с регистрацией у каждого тестового сообщения значений реального времени, когда его следует выдать на тестируемый компонент.

Для испытаний интегрированных проблемно-ориентированных АИС разработчики могут создавать программно-аппаратурные аналоги реальных объектов внешней среды для формирования части данных, а также ресурсы вычислительных средств для имитации данных от остальных объектов. Разумное сочетание части реальных объектов внешней среды и имитаторов на ЭВМ позволяет создавать высокоэффективные комплексные модели совокупностей объектов, необходимых для испытаний сложных АИС в реальном времени.

Такие модели позволяют осуществлять автоматическую генерацию тестов с помощью имитаторов на ЭВМ и аналогов реальной аппаратуры, дополнять данными от пользователей, контролирующих и корректирующих функционирование информационной системы.

Процессы создания тестов и обработки результатов организуются программно в моделирующей ЭВМ. Для каждого сеанса или эксперимента по испытаниям АИС реального времени подготавливаются план проверок и обобщенные исходные данные. Они включают:

- назначение испытания;
- область варьирования параметров;
- перечень контролируемых величин;
- метод получения эталонов;
- требуемые показатели качества и т.д.

Регистрация и обработка характеристик тестовых данных обеспечивает их контроль на соответствие заданным обобщенным характеристикам каждого объекта внешней среды и исходным данным сеанса испытаний. Часть этих характеристик используется для сопоставления с результатами функционирования испытываемого компонента для последующего определения некоторых показателей его качества.

Оперативная обработка результатов функционирования АИС обеспечивает замыкание обратной связи от проверяемых программ на имитаторы и реальные объекты внешней среды в реальном времени. Тем самым обеспечивается возможность испытаний АИС в контуре управления с учетом взаимодействия и реакции АИС и всех объектов внешней среды. На обработку этих результатов в моделирующей ЭВМ необходимо выделять некоторую долю производительности, достаточную для их полной обработки без искажения масштаба реального времени.

Часть результатов оперативной обработки используется только для подготовки данных при оценке качества АИС. Оценка качества заключается в:

- обобщении результатов тестирования;
- совместной обработке их с эталонными данными;
- расчете показателей качества, заданных техническим заданием или в спецификациях требований.

Расчеты показателей качества проводятся по методикам и алгоритмам, согласованным между разработчиками и заказчиком АИС. Эти методики уточняют и детализируют процессы обработки результатов испытаний и их сравнение с согласованными требованиями технического задания и проектно-конструкторской документации.

Важной функцией испытательных комплексов является возможность их использования в качестве тренажера для обучения пользователей до ввода АИС в эксплуатацию. Так как качество функционирования АИС может существенно зависеть от характеристик конкретного человека, участвующего в обработке информации, то необходимо измерять эти характеристики. Кроме того, необходимо иметь возможность их улучшать до уровня, обеспечивающего выполнение заданных требований. Поэтому в процесс испытаний может входить процесс тренировки и измерения характеристик реальной реакции пользователей. Следовательно, испытательный комплекс может служить прототипом для разработки тренажеров в серийных информационных системах с данными компонентами.

При испытаниях информационного компонента АИС (баз данных и файловых систем) на передний план выходит информация, подлежащая накоплению, хранению, обработке и использованию, но сохраняется достаточно важная роль ПС, реализующих процедуры обработки данных.

Таким образом, при анализе БД как объектов испытаний целесообразно рассматривать два компонента:

- ПС управления данными;

- совокупность данных, упорядоченных по некоторым правилам.

В распределенных, корпоративных АИС одна и та же система управления базой данных может обрабатывать различные по структуре, составу и содержанию данные, а одни и те же данные могут управляться программными средствами различных СУБД. Хотя эти компоненты тесно взаимодействуют при реализации конкретной прикладной БД, первоначально они создаются независимо и могут рассматриваться как *два объекта испытаний*, которые различаются:

- номенклатурой и содержанием показателей качества, определяющих их назначение, функции и потребительские свойства;

- технологией и средствами автоматизации разработки и испытаний объекта;

- эксплуатационной и технологической документацией, поддерживающей жизненный цикл объекта.

Кроме того, в больших проектах могут участвовать разные категории специалистов, обеспечивающих:

- создание БД (разработчики СУБД и разработчики исходных данных);

- эксплуатацию БД (администраторы СУБД и администраторы данных);

- применение БД (пользователи данных).

Эти обстоятельства позволяют подходить к сертификации БД первоначально как к испытаниям двух объектов с разными свойствами и испытывать их независимо. При этом СУБД может испытываться не на тех данных, которые впоследствии будут использоваться в конкретной прикладной области, а взаимодействие с конкретными проблемно-ориентированными данными может проверяться с использованием ПС на той СУБД, которая в дальнейшем будет применена в конкретной прикладной БД.

В конечном счете, пользователей интересуют совокупные характеристики качества конкретной используемой БД. Поэтому завершающие испытания и окончательная оценка БД должны проводиться для проверки функционирования и удостоверения показателей качества во взаимодействии с предполагаемой для использования СУБД, с вполне определенным наполнением базы данных.

В большинстве случаев важнейшими показателями качества СУБД являются функциональные характеристики процессов формирования и изменения информационного наполнения БД администраторами, а также доступа к данным и представления результатов пользователям БД.

Качество интерфейса пользователя с БД, обеспечиваемого средствами СУБД, оценивается, в значительной степени, субъективно, однако имеется ряд характеристик, которые можно оценивать достаточно корректно. В зависимости от конкретной проблемно-ориентированной области применения СУБД, приоритет при оценке качества может отдаваться:

- надежности и защищенности применения (финансовая сфера);
- удобству использования малоквалифицированными пользователями (социальная сфера);
- эффективности использования ресурсов (сфера производства).

Специфика испытаний СУБД, так же как и других типов ПС, сосредоточивается на выборе адекватных показателей качества из стандартной номенклатуры, на особенностях генерации тестов и обработке результатов тестирования. Так как поставщиками информации для СУБД чаще всего являются пользователи, они же должны выступать в роли генераторов тестов. Часть тестов может носить достаточно абстрактный характер и формироваться для заполнения БД специальными программами для испытания основных операций обработки данных. Часть функций, связанных с телекоммуникацией, должна испытываться на соответствие стандартам и протоколам телекоммуникации и взаимосвязи открытых систем.

Другая значительная часть функций непосредственно обусловлена спецификой применения распределенной СУБД. Некоторые из этих функций регламентируются специальными стандартами, в которых, в частности, представлены рекомендации по их аттестации. Кроме того, для распределенных СУБД значительно возрастает номенклатура сочетаний типов ЭВМ, выполняющих роль клиентов и серверов. Комбинаторика подобных типов ЭВМ и их операционных систем может быть весьма велика, и при испытаниях СУБД трудно охватить и проверить все особенности взаимодействия в таких распределенных СУБД. Поэтому протоколы испытаний распределенных СУБД должны отражать номенклатуру типов ЭВМ и операционных систем, для которых они предназначены и на которых испытаны.

Вторым компонентом для испытаний БД является собственно накапливаемая и обрабатываемая информация в базе данных. Показатели качества для БД значительно отличаются от применяемых при испытаниях ПС. Однако может сохраняться общий подход к определению и выделению адекватной номенклатуры показателей качества и их упорядочению. Он состоит в том, что выделяемые показатели качества должны иметь практический интерес для пользователей БД и быть упорядочены в соответствии с приоритетами практического применения. Кроме того, каждый выделяемый для проверки показатель должен быть пригоден для достаточно достоверного измерения и сравнения с требуемым значением при испытаниях.

Критерии качества БД рассмотрены в п.1.2.3. настоящей юниты.

По завершению комплексных испытаний АИС и ее компонентов, разработчик на основании полученных результатов должен принять решение о начале подготовки к приемо-сдаточным испытаниям. Эта подготовка заключается в разработке комплекта эксплуатационной документации для пользователей и документировании результатов предварительных испытаний, описаний и характеристик АИС для предъявления заказчику на приемо-сдаточные испытания. Приемо-сдаточные испытания заключаются в проведении тестирования АИС по программе приемо-сдаточных испытаний на соответствие функциональным и техническим характеристикам, заданным в контракте и согласованным с заказчиком.

Программа испытаний является планом проведения серии экспериментов и разрабатывается с позиции минимизации объема тестирования для проверки выполнения всех требований технического задания и сопровождающих документов. Программа испытаний должна содержать следующие основные, четко сформулированные, разделы:

объект испытаний, его назначение и перечень основных документов, определивших его разработку;

цель испытаний с указанием требований технического задания, параметров, подлежащих проверке и ограничений на проведение испытаний;

собственно программу испытаний, содержащую проверку комплектности предъявленной АИС в соответствии с технической документацией;

план тестирования для проверки по всем разделам технического задания и дополнительным требованиям, формализованным отдельными совместными решениями заказчика и разработчика;

методики испытаний, однозначно определяющие все понятия проверяемых характеристик и условия тестирования;

средства, используемые для автоматизации испытаний;

методики обработки и оценки результатов тестирования по каждому разделу программы испытаний.

ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

- 1. Составьте логическую схему базы знаний по теме юниты.*

Задание 2. В приведенном ниже абзаце вместо многоточия вставьте пропущенные слова:

Прототип АИС предназначен для, корректировки и с заказчиками решений проекта и требований на действующей модели АИС.

Задание 3. В приведенном ниже абзаце вместо многоточия вставьте пропущенные слова:

Имитация на прототипе - это метод проведения экспериментов с помощью прототипа АИС, моделирующий основные моменты реальной эксплуатации АИС в режимах:

..... **имитации** - имитационного эксперимента, позволяющего моделировать поведение АИС в определенный момент времени;

..... **имитации** - имитационного эксперимента, позволяющего моделировать поведение АИС в течение продолжительного периода времени.

Задание 4. Заполните левую сторону таблицы:

| Название термина | Определение термина |
|------------------|---|
| | описание возможных состояний связей, действующих между объектами в предметной области |
| | совокупность описаний объектов в предметной области и текущих состояний их связей |
| | описание связей в БД, с точки зрения пользователей и прикладных программ |

Задание 5. В таблице 1 приведено ненормализованное отношение СТУДЕНЧЕСКИЕ_ГРУППЫ. В таблицу 2 запишите нормализованное отношение СТУДЕНТЫ (как нормализованный вариант отношения СТУДЕНЧЕСКИЕ_ГРУППЫ)

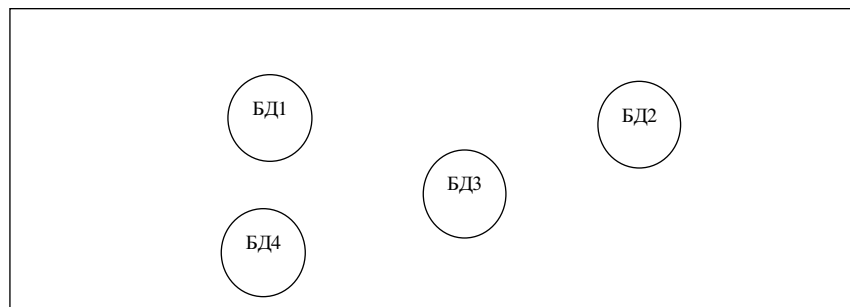
Таблица 1

| НОМЕР_ГРУППЫ | НОМ_СТУД_БИЛЕТА | ФАМИЛИЯ_СТУДЕНТА |
|--------------|-------------------------|--------------------------------|
| 34 | 12367 12378 12389 | Гольдберг Беляев Лосев |
| 35 | 23456 23467 23489 | Залоев Гусев Мирров |
| 36 | 32112 32132 32154 | Кучма Марлинский Федоров |

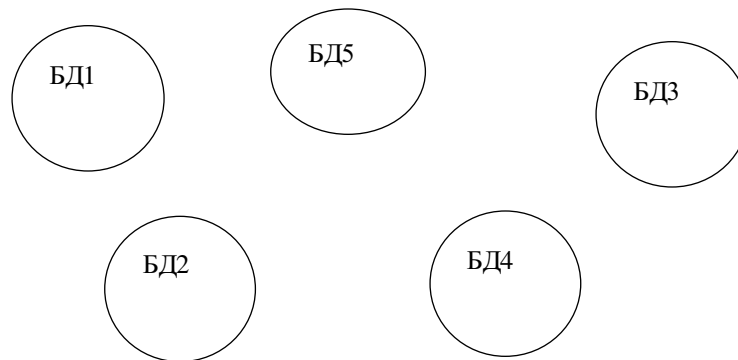
Таблица 2

| | | |
|--|--|--|
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Задание 6. Соедините базы данных, изображенные на рисунке, в радиально-кольцевую структуру (БД3 - центральная):



Задание 7. Соедините базы данных, изображенные на рисунке, в радиально-узловую структуру:



Задание 8. Заполните левую сторону таблицы соответствующим названием документа рабочего проекта:

| Название документа | Содержание документа |
|--------------------|--|
| | сведения о логической структуре и функционировании программы; |
| | схема алгоритма, общее описание алгоритма и функционирования программы, обоснование принятых технических решений; |
| | сведения, необходимые для установки, настройки и эксплуатации программы; |
| | программный документ, содержащий сведения для обеспечения процедуры общения с исполнительной системой в процессе выполнения программы. |



**ПРИНЦИПЫ ПОСТРОЕНИЯ
АВТОМАТИЗИРОВАННЫХ
ИНФОРМАЦИОННЫХ СИСТЕМ (АИС)**

ЮНИТА 3

ТЕХНИЧЕСКОЕ И РАБОЧЕЕ ПРОЕКТИРОВАНИЕ АИС

Редактор Л.С. Лебедева
Оператор компьютерной верстки Е.М. Кузнецова

| | |
|---|----------------|
| Изд. лиц. ЛР № 071765 от 07.12.1998 | Сдано в печать |
| НОУ “Современный Гуманитарный Институт” | |
| Тираж | Заказ |

Современный Гуманитарный Университет