Воронов М.П.
Кох Е.В.
Часовских В.П.
Анянова Е.В.

КОМПЬЮТЕРНЫЕ МЕТОДЫ ОБРАБОТКИ ИНФОРМАЦИИ В МЕНЕДЖМЕНТЕ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

УРАЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ЛЕСОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Воронов М.П., Кох Е.В., Часовских В.П., Анянова Е.В.

КОМПЬЮТЕРНЫЕ МЕТОДЫ ОБРАБОТКИ ИНФОРМАЦИИ В МЕНЕДЖМЕНТЕ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

Екатеринбург 2018

УДК 004.4'22

Рецензент:
Доросинский Леонид Григорьевич, профессор, доктор технических наук, директор департамента радиоэлектроники и связи Уральского федерального университета им. первого Президента России Б.Н. Ельцина

Воронов М.П., Кох Е.В., Часовских В.П., Анянова Е.В.
Компьютерные методы обработки информации в менеджменте лесопромышленного предприятия. Учебное пособие. – Екатеринбург: Уральский государственный лесотехнический университет, 2018. 303 с.

ISBN 978-5-94984-640-7

Учебное пособие посвящено методам моделирования информационных систем поддержки принятия решений с учетом специфики лесопромышленных предприятий. Рассматриваются теоретические и практические аспекты организации и средств информационных технологий управленческой деятельности; информационных технологий документационного обеспечения управленческой деятельности; информационного обслуживания управленческой деятельности; основы построения инструментальных средств корпоративной информационной системы лесопромышленного предприятия.

Приводится описание основных структур баз данных, вопросы проектирования и эксплуатации информационных систем управления лесопромышленными предприятиями, аспекты создания пользовательских приложений в среде ADABAS и Natural.

Для студентов направления 09.03.02 - прикладная информатика, 27.03.02 - управление качеством.

Печатается по решению редакционно-издательского совета
Уральского государственного лесотехнического университета

ISBN 978-5-94984-640-7

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 6
1. СПЕЦИФИКА КОРПОРАТИВНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ 10
1.1. Структура и модульный состав корпоративной информационной системы лесопромышленного предприятия 10
1.2. Специфика исходных данных информационной системы управления лесопромышленного предприятия 24
1.2.1. Механизированная и машинная валка деревьев 28
1.2.2. Технологические процессы трелевки древесины 29
1.2.3. Очистка деревьев от сучьев 29
1.2.4. Погрузка древесины на верхних складах 30
1.2.5. Заготовка сортиментов на лесосеке 30
1.2.6. Лесопильное производство 32
1.2.7. Фанерное производство 38
1.2.8. Производство древесных плит 41
1.2.9. Мебельное производство 43
1.2.10. Вспомогательные подразделения 44
1.3. Методы доступа к БД информационной системы управления лесопромышленного предприятия 44
1.4. Выводы 46
2. СПЕЦИФИКА СРЕДЫ РЕАЛИЗАЦИИ ИНФОРМАЦИОННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ 47
2.1. Специфика СУБД ADABAS 48
2.1.1. Модель и структура данных 49
2.1.2. Структура хранения данных 53
2.1.3. Методы доступа к данным 59
2.1.4. Ограничения целостности 65
2.2. Специфика среды проектирования приложений Natural 67
2.2.1. SPoD - единая система разработки прикладных приложений 68
2.2.2. Средства взаимодействия приложений системы с СУБД ADABAS 72
2.2.3. Графический интерфейс и обработка отчетных форм 73
2.3. Функционирование системы в условиях распределенной среды 75
2.4. Стоимость ПО среды проектирования и эксплуатации 78
2.5. Выводы 79
3. СИСТЕМНЫЕ СВЯЗИ И ФУНКЦИОНИРОВАНИЕ КОМПОНЕНТОВ КОРПОРАТИВНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ 81
3.1. Уровень организации ядра КИС лесопромышленного предприятия 81
3.1.1. Основные аспекты организации ядра в СУБД ADABAS 82
3.1.2. Реляционная структура ядра 83
3.1.3. Иерархическая структура ядра 86
3.1.4. Многоуровневая структура ядра 88
3.1.5. Мультипольная структура ядра 90
3.1.6. Смешанная структура ядра 92
3.1.7. Рекомендации по выбору структуры ядра для КИС лесопромышленного предприятия 94
3.2. Уровень пользовательских приложений 101
3.2.1. Формирование списка активных элементов пользовательских приложений 101
3.2.2. Формирование списка активных пользовательских приложений 102
3.2.3. Автоматическая настройка активных пользовательских приложений 110
3.3. Уровень расчетных программ КИС 114
3.3.1. Оптимизация запросов 114
3.3.2. Использование различных типов переменных 119

3.3.3. Формирование списка активных расчетных программ ... 126
3.3.4. Использование системы индикаторов расчетными программами ... 128
3.4. Уровень отчетных форм ... 132
3.4.1. Получение отчетных форм на основе задаваемых критериев ... 132
3.4.2. Формирование списка активных отчетных форм ... 133
3.4.3. Автоматическая настройка элементов отчетных форм ... 136
3.5. Выводы ... 139
4. АНАЛИЗ ЭФФЕКТИВНОСТИ СИСТЕМНЫХ СВЯЗЕЙ И ЗАКОНОМЕРНОСТЕЙ ФУНКЦИОНИРОВАНИЯ КИС ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ ... 142
4.1. Основные показатели эффективности различных способов организации ядра КИС лесопромышленного предприятия ... 142
4.1.1. Реляционная структура ядра ... 142
4.1.2. Иерархическая структура ядра ... 143
4.1.3. Многоуровневая структура ядра ... 144
4.1.4. Мультипольная структура ядра ... 145
4.2. Анализ экономической эффективности внедрения разработанных компонентов и системных связей в КИС лесопромышленного предприятия ... 145
4.3. Выводы ... 152
5. ОПРЕДЕЛЕНИЕ ЛОГИЧЕСКОЙ И ФИЗИЧЕСКОЙ СТРУКТУРЫ БД ADABAS ... 154
5.1. Определение БД ... 154
5.2. Определение файла БД ... 161
4.3. Определение записи ... 166
4.4. Определение параметров ассоциатора и обработки данных ... 197
6. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЙ КИС ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ В СРЕДЕ NATURAL ... 205
6.1. Основные возможности и компоненты редактора Natural ... 205
7.2. Определение локальной области данных ... 219
6.3. Создание DDM файла БД ... 230
6.4. Редактирование программ Natural ... 239
6.5. Экранные формы ввода ... 249
6.6. Свойства и возможности диалогов Natural ... 262
7. СОСТАВ И ИНСТРУМЕНТАРИЙ ЕДИНОЙ СРЕДЫ РАЗРАБОТКИ ПРИЛОЖЕНИЙ SPOD ... 276
7.1. Архитектура среды SPoD ... 276
7.2. Функции и свойства среды SPoD ... 278
7.3. Компоненты среды SPoD ... 279
БИБЛИОГРАФИЧЕСКИЙ СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 282

ВВЕДЕНИЕ

В настоящее время в условиях возрастающей конкуренции возможность гибкого реагирования на изменения условий рынка и оптимального распределения производственных ресурсов приобретает все большую значимость для предприятий российской промышленности. В XXI-ом веке в управлении организацией доминируют информация и технологии. Именно информация и управление человеческими знаниями в настоящее время определяют власть в организации. Управление знаниями становится важнейшим фактором создания благ и обеспечивает конкурентные преимущества лишь в том случае, если оно рассматривается не в качестве структурного звена, а понимается и формируется как инструмент управления организацией.

При создании современных систем принятия решений, и в условиях постоянного повышения качества принимаемых решений, автоматизированные системы управления предприятий базируются на все более совершенных математических и информационных моделях, и все более обширная сфера факторов влияния используется в качестве исходных данных анализа. Современные интеллектуальные системы принятия решений на основе анализа прошлых тенденций и анализа эффекта от уже принятых решений, ставят перед необходимостью хранения и обработки все больших объемов данных.

В условиях конкуренции, возможность быстрого реагирования на изменения условий внешней среды, проведения оперативного анализа и своевременного принятия решений, представляет для организации все больший интерес. И предприятие, имеющее преимущества в скорости обработки данных обладает существенным конкурентным преимуществом. Таким образом, существует необходимость постоянного повышения эффективности использования информационных систем, обрабатывающих большие массивы данных. В данном разрезе исследование системных связей и функциональных закономерностей информационных систем, как средств повышения

производительности обработки информации (и как следствие, повышение конкурентоспособности предприятия) представляют широкий практический и научный интерес.

Основу любой автоматизированной системы управления производством (АСУП), а также ее наиболее современной модификации - корпоративной информационной системы (КИС), составляют ее информационные модели и средства их обработки, представленные в совокупности баз данных (БД).

Применение СУБД позволяет существенно повысить надежность и эффективность обработки информации в сложных информационных системах, сократить сроки и затраты на их проектирование, внедрение и эксплуатацию. Еще большая эффективность от применения функционально развитых СУБД может быть получена на этапах их внедрения и эксплуатации в распределенных системах обработки данных за счет использования типовых проектных решений для узлов сети и централизации всех этапов жизненного цикла СУБД. При создании корпоративных информационных систем оптимизация процедур обмена данными в распределенной среде приобретает все большую актуальность.

Немаловажным фактором, определяющим эффективность КИС является выбор СУБД и среды разработки и функционирования программных элементов информационной системы. Характерной чертой современных СУБД является их ориентация на решение прикладных задач, требующих возможность нестандартной обработки данных, а также возможность изменения пользователем требований к прикладным задачам в отношении способа обработки данных, структуры связей между объектами и выходных форм отчета. Построение КИС на основе современных СУБД осуществляется с помощью высокоуровневых, интегрированных с базой данных языков программирования. При этом эффективность сообщения БД со средой разработки информационной системы во многом определяется возможностями этих языков.

В рамках данной работы было принято решение остановить выбор на среде разработки СУБД ADABAS и редактора Natural, как одной из самых эффективных сред, существующих в мире.

СУБД ADABAS (Software AG, Германия) является профессиональной промышленной СУБД, предназначенной для создания ИС и решающей ряд трудноформализуемых прикладных задач. Это многофункциональная СУБД, с успехом применяемая в таких областях деятельности, как управление организацией, обработка научно-технической и библиографической информации, автоматизация проектных работ, обработка экономической информации. Она обеспечивает высокую производительность при работе с большими и сверхбольшими базами данных, обладает развитыми средствами контроля, поддержания и восстановления целостности баз данных.

Natural - платформа Software AG, предназначенная для разработки как транзакционных приложений, так и целых информационных систем. Это высокоуровневый язык программирования, позволяющий существенно сократить сроки и стоимость разработки бизнес-приложений, ограждая разработчиков от сложностей программирования. Natural поддерживает все типы пользовательских интерфейсов, включая Web, графический интерфейс Windows, текстовые терминалы. Приложения, написанные на Natural, могут быть легко интегрированы с любыми внешними сервисами, будь-то XML/Web-сервисы, DCOM, CORBA и др. Приложения на Natural могут работать с большинством реляционных и постреляционных СУБД. Среда разработки и исполнения Natural-приложений существует для всех основных операционных систем и аппаратных платформ, включая мэйнфреймы, Unix, Linux и OpenVMS.

Принимая во внимание, что вопросы создания КИС лесопромышленного предприятия в настоящее время недостаточно изучены и слабо освещены в литературе, необходимость их дальнейшей углубленной проработки и исследований представляется очевидной. Исследование системных связей КИС лесопромышленного предприятия является с одной стороны перспективным

направлением снижения затрат на внедрение и эксплуатацию КИС, и с другой стороны способствует оптимизации модульных структур КИС и адаптации их к нуждам предприятия с учетом отраслевой и структурной специфики, структурных, производственных и качественных изменений в деятельности предприятия.

Таким образом, данная работа посвящена развитию и конкретизации возможных подходов к созданию и эксплуатации КИС лесопромышленного предприятия в соответствии с особенностями его структуры, используемых технологий и видов производств.

Учебное пособие предназначена для студентов очной и заочной форм обучения по направлению направлению 09.03.02 - прикладная информатика, 27.03.02 - управление качеством..

1. СПЕЦИФИКА КОРПОРАТИВНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

1.1. Структура и модульный состав корпоративной информационной системы лесопромышленного предприятия

Корпоративная информационная система (КИС) рассматривается в виде совокупности программных модулей, каждый из которых осуществляет управление конкретной сферой деятельностью организации. Программные модули также включают концепции и методологии планирования и управления в различных сферах деятельности организации. Выбор состава модулей в рамках информационной системы зависит от размеров организации, ее технологических особенностей, сферы ее деятельности, особенности организации сбыта продукции и т.д.

При создании КИС промышленного предприятия наиболее предпочтительной концепцией является ERP, т.к. позволяет планировать все виды ресурсов предприятия.

Согласно описаниям, встречающимся в литературе [6, 7, 24, 58, 62, 100, 101, 103, 104] КИС промышленного предприятия, разрабатываемая в соответствии с концепцией ERP, должна состоять из следующих функциональных модулей:

1. Программный модуль управления производством (включает управление лесопильным производством, производством древесных плит, мебельным производством и других блоков управления, в зависимости от специфики предприятия).
2. Программный модуль управления производственными запасами.
3. Программный модуль управления сбытом готовой продукции.
4. Программный модуль управления учетной деятельностью.
5. Программный модуль управления планово-аналитической деятельностью.

Управление производственными запасами подразумевает формирование оптимального объема запасов предприятия, достаточного для обеспечения процесса производства в объеме соответствующим потребностям рынка. Т.к. излишки производственных запасов «замораживают» оборотный капитал организации, а также увеличивают затраты на хранение, объем запасов должен быть всегда регламентированным в соответствии с уровнем спроса за прогнозируемый период [113, 114].

Также следует учитывать факт, что в связи с сезонными колебаниями уровня цен на некоторые группы материалов в целях снижения общих затрат целесообразно проводить закупки товара в периоды, когда цена является наименьшей. При этом неотъемлемым условием формирования дополнительных запасов является:

$$\Delta_i > C_i + E_i$$, где

Δi – изменение цены единицы i-го материала за счет фактора сезонности,

Ci – стоимость хранения единицы i-го материала за период хранения,

Ei – альтернативные издержки, прибыль, которая могла быть получена при использовании суммы, затраченной на приобретение единицы i-го товара в прочих видах деятельности предприятия.

Потребность организации в тех или иных запасах рассчитывается на основе анализа спроса [114]. Также на основе объема спроса и нормативных данных расхода материалов на производство единицы изделия рассчитываются объемы запасов всех видов материалов, и полученные результаты заносятся в БД в соответствии с номенклатурой материалов.

Номенклатура требуемых материалов составляется в соответствии с потребностями организации по одному из описанных ниже видов модели структуры.

Далее составляется план проведения закупок материалов (автоматический вывод формы). По мере проведения закупок фактические объемы каждого из

видов материалов сравниваются с запланированными, при этом данные об объемах заносятся в БД.

При проведении закупок одного вида товаров несколькими партиями данные об объеме запаса суммируются, и при превышении фактического объема запаса материалов над запланированным значением, выдается сообщение о превышении объема запаса и сумма дополнительных затрат, связанных с хранением материалов и «заморозкой» капитала.

Модуль управления производственными запасами может быть отображен в виде следующей схемы (рис 1.1.).

На рис. 1.1. введены следующие условные обозначения:

1 – данные за предыдущий период.

2 – результаты анализа хозяйственной деятельности; результаты прогноза объема спроса.

3 – результаты анализа хозяйственной деятельности; результаты прогноза объема спроса; нормы расхода материалов на производство единицы продукции; данные о материалах, цены на которые подвержены сезонным колебаниям.

4 – результаты планирования производственных запасов.

5 – результаты планирования производственных запасов.

6 – фактические данные о формировании производственных запасов.

7 – данные о дополнительных затратах.

Основными функциями планово-аналитического модуля являются:

1. Выявление данных о прогнозируемом объеме продаж на основе:
    a. Данных об объемах продаж за предыдущие периоды.
    b. Тенденций развития рынка (к понижению или повышению спроса на данную продукцию).
2. Анализ хозяйственной деятельности.

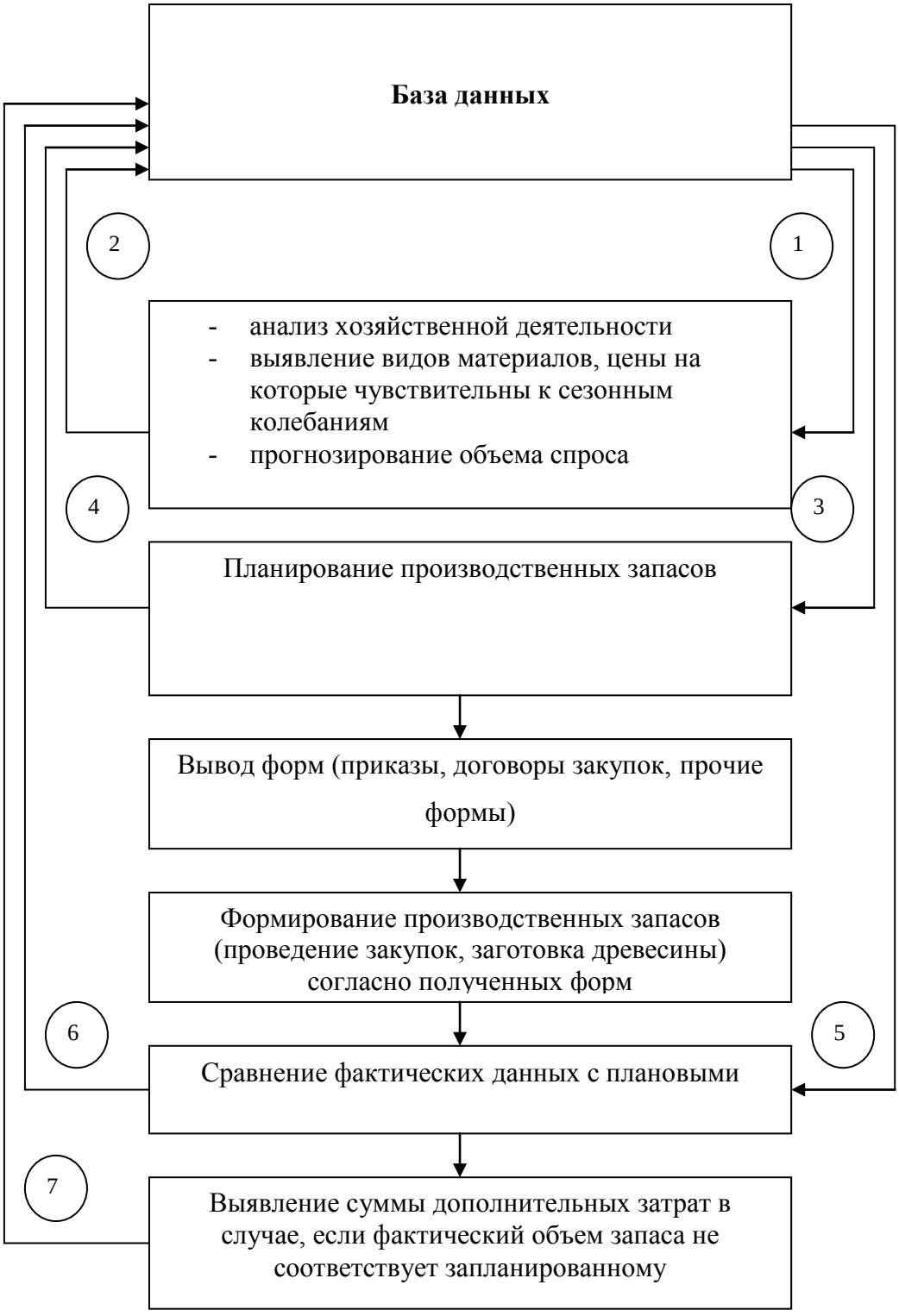


Рис 1.1. Схема модуля управления производственными запасами

3. Составляет планы на отчетный период в соответствии со стратегическими целями и проведенного анализа хозяйственной деятельности по видам деятельности:

a. Планирование производственных запасов.

b. Планирование фонда заработной платы.

c. Планирование численности персонала и прочие.

Прогнозирование объема продаж проводится по следующим этапам:

1. Формирование запросов данных за предшествующие периоды к БД.

2. Формализация обработки запрашиваемых данных в соответствии с принятыми на предприятии статистическими и математическими методами.

3. Вывод данных в виде стандартизованных форм и предоставление обработанных данных для использования в прочих видах деятельности предприятия.

Модуль управления планово-аналитической деятельностью представлен в виде схемы (рис.1.2.).

На рис. 1.2. введены следующие условные обозначения:

1 – данные за предыдущий период деятельности.

2 – результаты проведения анализа хозяйственной деятельности.

3 – данные о состояниях внешней среды предприятия за предыдущие периоды.

4 – результаты проведения анализа внешней среды предприятия.

5 – результаты проведения анализа хозяйственной деятельности.

6 – результаты планирования.

Создание модуля управления учетной деятельностью подразумевает создание системы, позволяющих пользователям вводить данные в БД, а затем, на основе автоматических расчетов требуемых величин, получать формы первичной документации, а также, отчетные формы [53].

Модуль управления производством отражен в виде схемы (рис 1.3.).

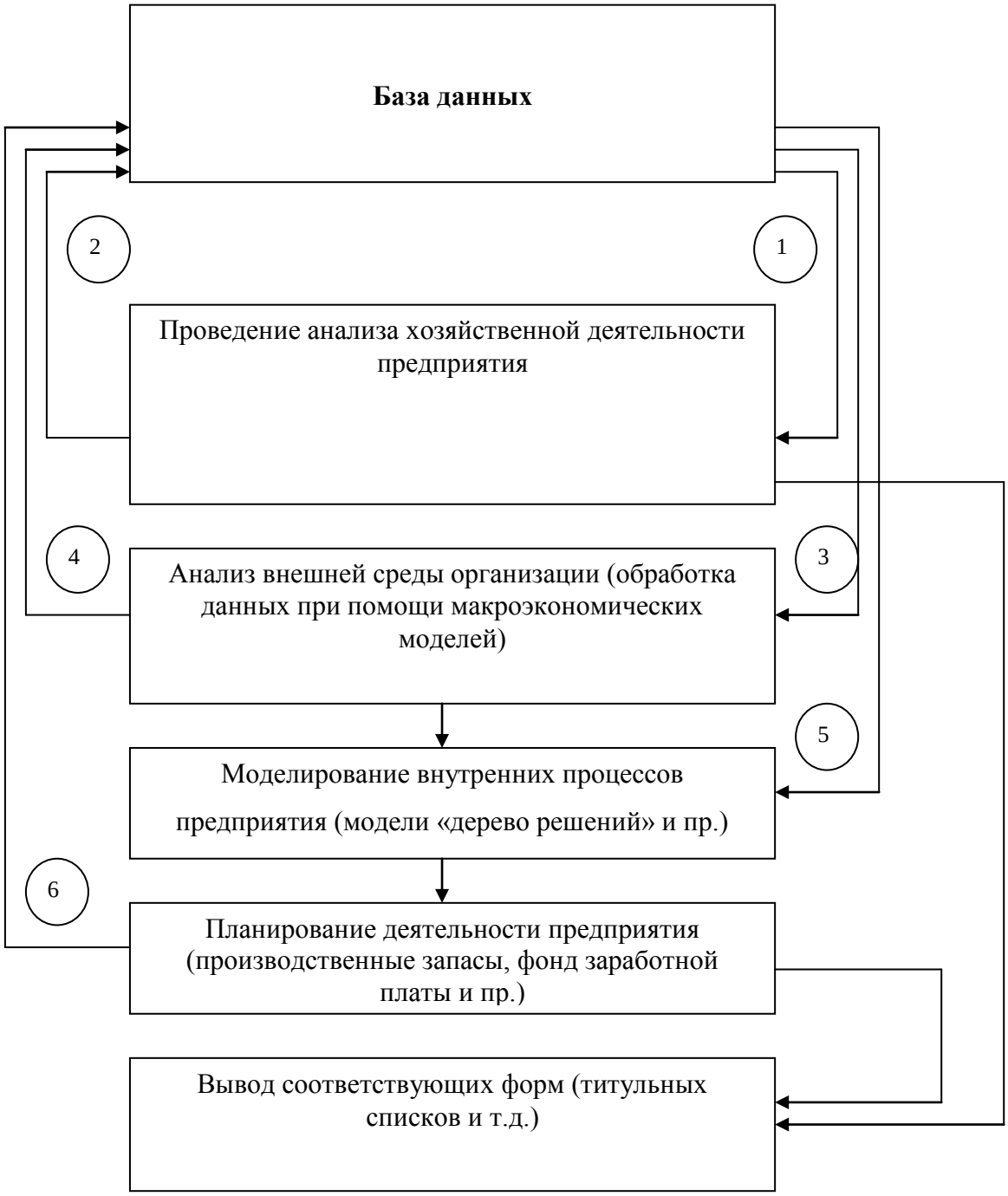


Рис 1.2. Схема управления планово-аналитической деятельностью

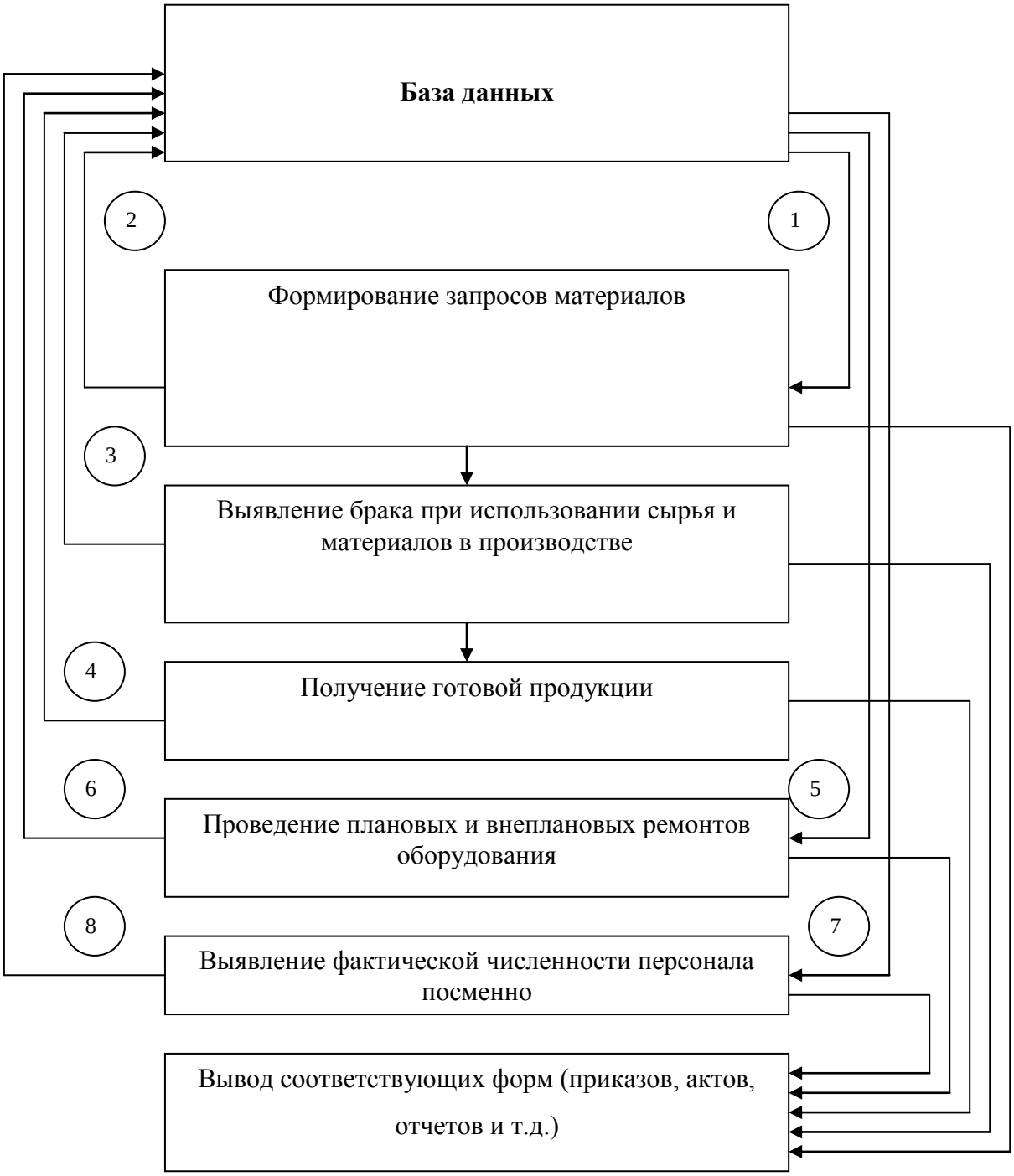


Рис 1.3. Схема модуля управления производством

На рис. 1.3. введены следующие обозначения:

1 – данные об объеме производственных запасов.

2 – данные об объеме материалов, необходимых в производстве.

3 – данные о видах и объеме бракованных материалов.

4 – данные о полученной готовой продукции.

5 – данные о уже проводимых ремонтах оборудования.

6 – данные о проводимых ремонтах оборудования.

7 – данные о численности персонала посменно.

8 – данные о фактической численности персонала посменно.

При осуществлении сбыта продукции наряду с маркетинговой деятельностью осуществляется выработка плана поставок на основе данных о потребителях продукции предприятия и данных о наличии готовой продукции. Схема управления маркетинговой деятельностью схожа со схемой управления планово-аналитической деятельностью (рис 1.2.).

Схема модуля управления сбытом готовой продукции показана на рис 1.4.

На рис. 1.4. введены следующие обозначения:

1 – данные о потребителях продукции предприятия и о готовой продукции.

2 – результаты планирования поставок.

3 – результаты планирования поставок.

4 – фактические данные о проведении поставок.

5 – данные о излишке или недостатке готовой продукции и связанных с ним убытков.

При помощи программного модуля управления учетной деятельностью осуществляется структурирование, хранение, редактирование и интерпретация данных различных сфер деятельности предприятия в соответствии со схемами на рис. 1.1., 1.2., 1.3., 1.4.

Всеми программными модулями КИС лесопромышленного предприятия выполняются следующие функции:

1. Планирование продаж и производства. Результатом действия блока является разработка плана производства основных видов продукции.

2. Управление спросом. Данный блок предназначен для прогноза будущего спроса на продукцию, определения объема заказов, которые можно предложить клиенту, определения спроса дистрибьюторов и др.

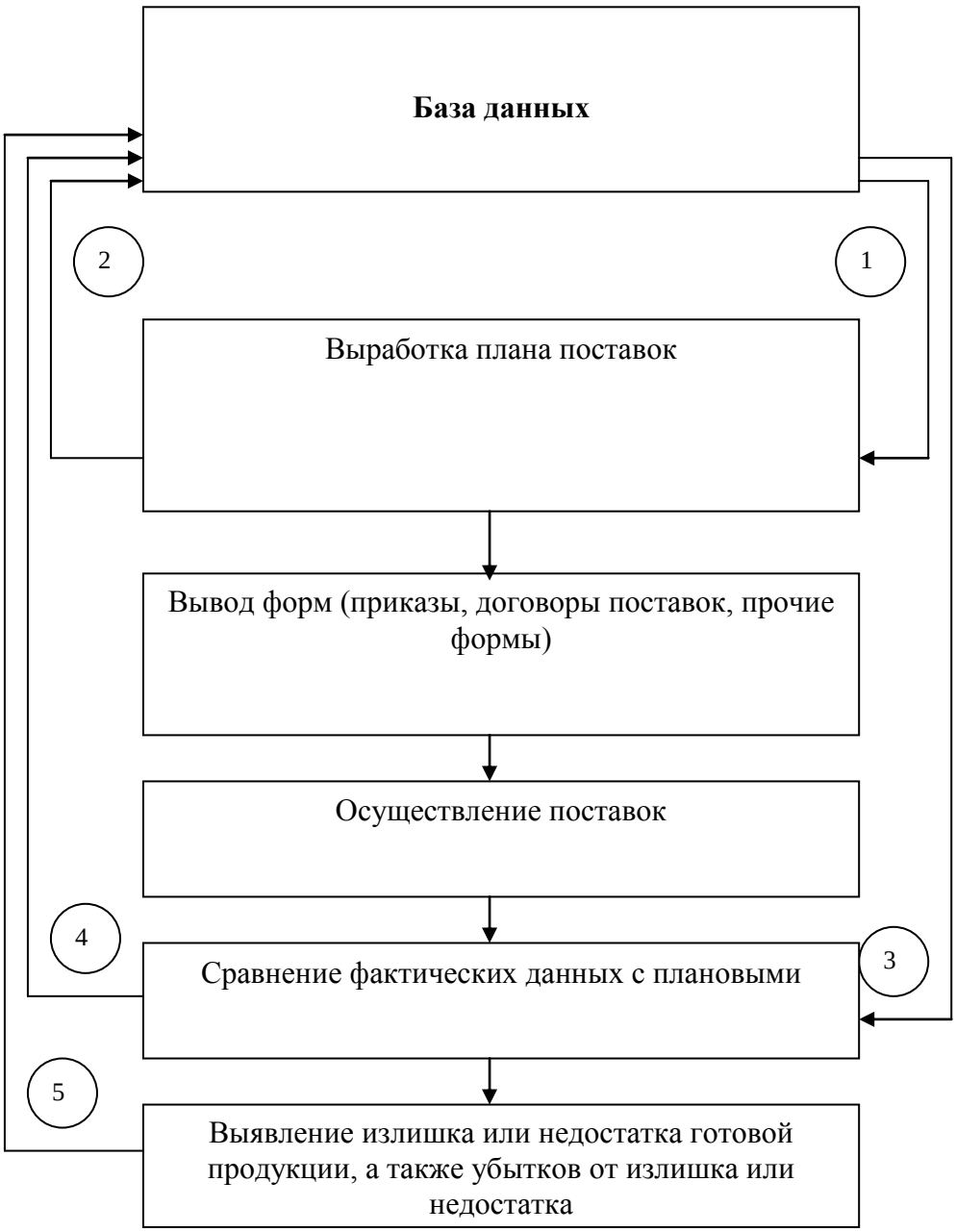


Рис 1.4. Схема модуля управления сбытом готовой продукции

3. Укрупненное планирование мощностей. Используется для конкретизации планов производства и определения степени их выполнимости.

4. Основной план производства (план-график выпуска продукции). Определяется продукция в конечных единицах (изделиях) со сроками изготовления и количеством.

5. Планирование потребностей в материалах. Определяются виды материальных ресурсов (сборных узлов, готовых агрегатов, покупных изделий, исходного сырья, полуфабрикатов и др.) и конкретные сроки их поставки для выполнения плана.

6. Спецификация изделий. Определяет состав конечного изделия, материальные ресурсы, необходимые для его изготовления, и др. Фактически спецификация является связующим звеном между основным планом производства и планом потребностей в материалах.

7. Планирование потребностей в мощностях. На данном этапе планирования более детально, чем на предыдущих уровнях, определяются производственные мощности.

8. Маршрутизация/рабочие центры. С помощью данного блока конкретизируются как производственные мощности различного уровня, так и маршруты, в соответствии с которыми выпускаются изделия.

9. Проверка и корректировка цеховых планов по мощностям.

10. Управление закупками, запасами, продажами.

11. Управление финансами (ведение Главной книги, расчеты с дебиторами и кредиторами, учет основных средств, управление наличными средствами, планирование финансовой деятельности и др.).

12. Управление затратами (учет всех затрат предприятия и калькуляция себестоимости готовой продукции или услуг).

В общем виде, модель КИС промышленного предприятия (а также любой из программных модулей КИС) может быть представлена в виде схемы (рис. 1.5.).

На рис. 1.5. введены следующие условные обозначения:

1 – обращение к программам управления данными (запуск программ).

2 – запросы данных, проведение выборки данных, внесение данных и пр.

3 – данные.

4 – данные, результаты выборки, результаты.

5 – обращение к программам расчета величин и показателей на основе полученных данных (запуск программ).

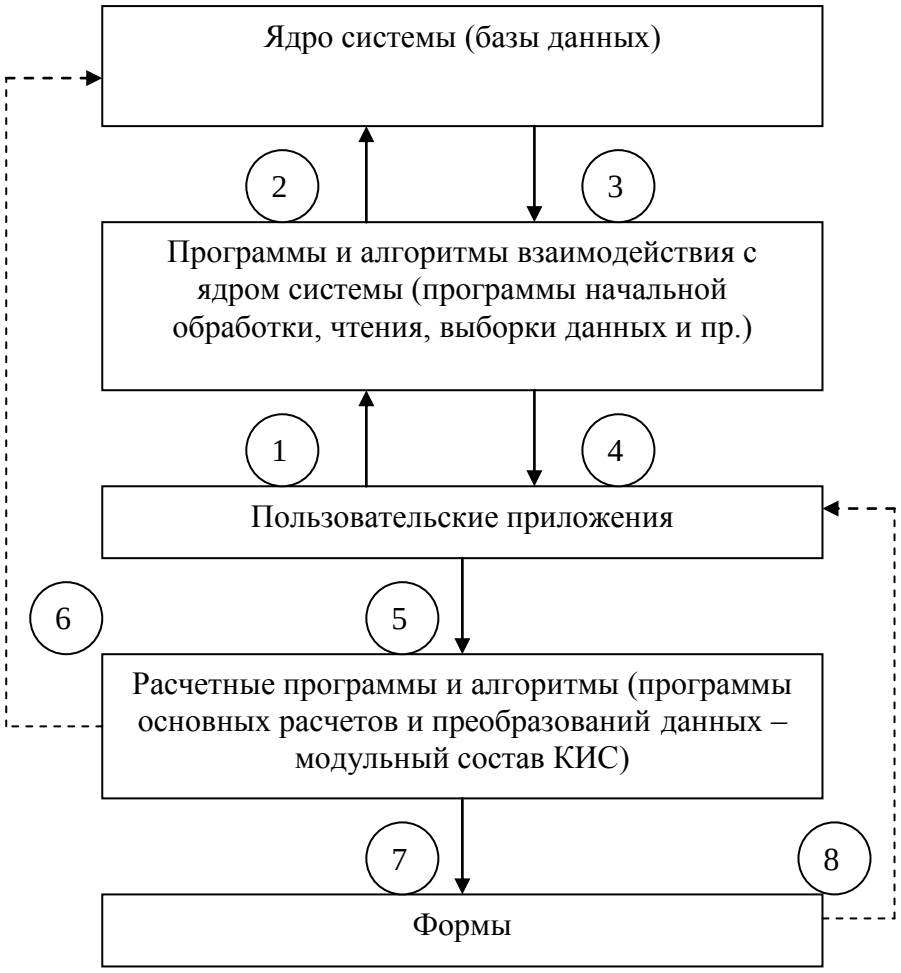


Рис 1.5. Схема КИС промышленного предприятия (программных модулей КИС промышленного предприятия)

6 – результаты проведения анализа.

7 – представление результатов расчетов в виде форм, результаты.

8 – формы.

Далее приведено описание компонентов процесса обработки данных в рамках КИС промышленного предприятия [58]:

Ядро системы (базы данных). В файлах структуры БД фиксируются структура хранения данных и свойства обработки для каждой единицы данных (для каждого поля данных). В файлах структуры БД могут быть отражены следующие составляющие системы:

−система критериев оценки конкурентных преимуществ, и состояния внешней среды;

−показатели и индикаторы, на основе которых производится анализ и расчеты;

−система показателей, используемая в производственных процессах предприятия (в т.ч. нормативы, коэффициенты, шифры, ставки и т.д.);

−внутренние показатели организации (сведения о сотрудниках, произведенной продукции, данные о затратах и пр.);

−результаты вычислений, произведенных в результате процесса принятия решений на основе внутренних стандартов, показателей и индикаторов;

−прочие.

Программы и алгоритмы взаимодействия с ядром системы. Основное назначение данных программ – осуществление управления БД. Все программы и алгоритмы можно подразделить на следующие группы:

−программы первичной обработки данных и ввода данных в БД (в т.ч. программы проверки вводимых данных на соответствие);

−программы, осуществляющие чтение данных в БД в заданной последовательности;

−программы, осуществляющие выборку данных из БД по заданным параметрам (критериям);

−программы интерпретации показателей и индикаторов;

−программы расчета значений на основе запрашиваемых данных.

Все программы и алгоритмы представляют собой программный код. В целях избежания ошибок при чтении и вставке данных, необходимо создание программных кодов всех видов запросов к БД и их последующая

стандартизация, т.е. установление внутриорганизационных стандартов для процедур получения и вставки данных с учетом периодов времени, за которые запрашиваются (вставляются) данные, типов данных и пр. При этом отпадает необходимость написания программы-запроса при создании пользовательских приложений, вместо чего вызывается уже стандартизированная программа, которая предоставляет обработанные данные в окно диалога.

Пользовательские приложения. Назначение пользовательских приложений в рамках КИС промышленного предприятия – управление данными. Пользовательские приложения обеспечивают сотрудников предприятия возможностями:

–Получения данных и результатов выборки за различные периоды времени. Периоды времени задаются ввода соответствующего параметра, который передается программе, исполняющей выборку в соответствии со значениями параметра.

–Ввода данных для внесения в БД и дальнейшей обработки. В данном случае приложение должно предоставлять пользователю для заполнения форму, содержащую поля ввода данных. По окончании заполнения пользователем формы вызывается исполняющая программа, которая обеспечивает ввод данных в БД. В целях избежания ошибок ввода рекомендуется предварительная проверка исполняющей программой введенных данных на соответствие требованиям к формату данных перед осуществлением процедуры ввода данных в БД.

–Редактирования ранее введенных данных. Как правило, подразумевается следующая последовательность: чтение (выборка) данных из БД; обработка полученных данных в окне пользовательского приложения; вставка данных в БД. Данная процедура представляет собой совокупность двух вышеперечисленных.

−Удаление ранее вводимых данных. В целях обеспечения возможности восстановления данных в случае непроизвольного удаления данных пользователями рекомендуется следующая организация процедуры удаления:

−Создание однобайтового поля индикатора активности учетной записи при проектировании структуры БД (таблица FDT).

−Чтение данных последовательно, по каждой учетной записи, подлежащей удалению и вывод данных в окне пользовательского приложения.

−Присвоение полю индикатора активности каждой удаляемой учетной записи значения, соответствующего неактивному статусу учетной записи по команде пользователя, исходящей из приложения.

−Вставка данных в БД.

−Редактирование текста внутри окна пользовательского приложения.

−Отображение полученных данных в единой форме и возможность манипулирование данными, а также возможность вызова программ дальнейшего расчета для показателей, указанных посредством окна приложения.

−Отображение и печать форм, сформированных после проведения основных расчетов.

Расчетные программы и алгоритмы. Расчетные программы и алгоритмы осуществляют основные расчеты и преобразования и образуют модульный состав КИС промышленного предприятия. Математические и статистические методы, оптимизационные модели, модель «дерево решений» и пр. могут реализовываться в рамках программных модулей программистами самого предприятия, а также приобретаться у разработчиков программных и расчетных модулей (в т.ч. интегрированные пакеты SAS) [98]. Рассчитанные величины при необходимости окончательно интерпретируются в соответствии с принятыми на предприятии стандартами, и передаются в формы.

Формы. Все результаты расчетов должны отображаться в виде стандартных форм (в соответствии ГОСТам, либо внутренним стандартам, в зависимости от назначения).

1.2. Специфика исходных данных информационной системы управления лесопромышленного предприятия

Определим специфику учетных данных в КИС предприятий лесной промышленности. Теория современного учета содержит основные правила логических и вычислительных действий над исходной и промежуточной информацией, преобразуемой учетом в интересах получения наиболее точных базисных (плановых, нормативных), контрольных (расчетных) и отчетных (результатных) показателей деятельности предприятия при фактических параметрах производственных и хозяйственных процессов [49, 71, 72, 78, 81, 86, 90, 91]. Теория формирования учетных записей БД лесопромышленного предприятия представлена в [115-117].

Определяющим фактором при создании ядра КИС лесопромышленного предприятия, а также программ и алгоритмов обработки данных являются специфика типов данных деревообрабатывающих производств, а также особенности учета в данной отрасли и типы отчетных форм.

Любую запись с одним (стоимостным) основанием можно представить как учетную фразу Mi:

Mi=θ;ψ;Db;Kr;P

где θ - технологическая характеристика записи;

ψ - обозначение признаков документа;

Db и Kr – дебет и кредит записи, т.е. шифры соответственно дебетуемых или кредитуемых счетов;

P – стоимостная характеристика сальдо или операции.

В СБЗ с двумя основаниями добавляется идентификатор натуральной характеристики H, так что учетная фраза Mi имеет вид

Mi=θ;ψ;Db;Kr;P;H.

Технологическая характеристика записи θ представляется в виде:

θ=Dp;Wt;It

где Dp - шифр подразделения;

Wt - шифр вида учетной работы;

It - шифр характера информации (сальдо дебета или кредита, прямая или сторнировочная проводка, итоговая запись, норматив).

Обозначение признаков документа ψ представляется в виде:

ψ=DD;MM;YYYY;Num

где DD – число;

MM – месяц;

YYYY – год;

Num – номер первичного документа.

Дебет (Db) и кредит (Kr) записи представляются в виде:

Db=dSinS;dSs;dAs

Kr=kSinS;kSs;kAs

где SinS1 - синтетический счет 1-го порядка;

Ss – субсчет;

As - аналитический счет (первые буквы d или k в названиях обозначают принадлежность соответственно к дебету или кредиту).

При необходимости могут использоваться аналитические счета второй, третьей ступеней и т.д.

В развернутом виде учетная фраза Mi имеет вид:

Mi=Dp;Wt;It;DD;MM;YYYY;Num;dSinS;dSs;dAs;kSinS;kSs;kAs;P

для СБЗ с одним основанием и

Mi=Dp;Wt;It;DD;MM;YYYY;Num;dSinS;dSs;dAs;kSinS;kSs;kAs;P;H

для СБЗ с двумя основаниями.

Таким образом, при помощи СБЗ, в каждой из которых содержатся данные для отнесения операции на соответствующий счет, фиксируется информация о затратах на производство.

Кроме хранилища учетных данных в КИС лесопромышленного предприятия необходимо предусматривать хранилище данных для плановых величин.

В процессе эксплуатации КИС расчетными программами и алгоритмами на основе оптовых и планово-расчетных цен, ценностных коэффициентов разновидностей пиломатериалов, комбинации коэффициентов сортности фанерных изделий, коэффициентов для вычисления нормированного расхода сырья, коэффициентов перевода фактического количества разнородных лесоматериалов и заготовок из древесины в условный материал производится расчет обобщающих величин. После чего фактически полученные результаты сравниваются с плановыми значениями, интерпретируются и представляются в виде отчетных форм.

Идентификаторы элементов учетных записей, формируемых по хозяйственным операциям, приведены в табл. 1.1. Также в табл. 1.1. для каждого параметра приведено количество возможных значений.

Рассмотрим специфику формирования и примеры учетных записей для нескольких видов производств деревообрабатывающей промышленности.

Переменные элементов записей производственного учета

Таблица 1.1.

Элементы записей	Переменные	Количество значений
1	2	3
Структурное подразделение предприятия	ξ	1…15
Производственный поток, стадия производства	ПП	1…6
Хозрасчетная бригада, участок	e	1…12

Номер смены (в цехе, отделении)	I	1…3
Калькулируемый объект (изделие, сортимент)	B	1…20
Наименование объекта (изделия, сортимента)	$B_{ДЕТ}$	1…30
Группа лесов	F	1…3
Категория рубки	R	1…10
Тип транспорта	T_s	1…20
Тип технологического процесса лесозаготовки	T_t	1…4
Схема разработки лесосеки	S_f	1…5
Схема разработки пасеки	S_p	1…5
Тип валочной машины	M_v	1…15
Способ трелевки	S_t	1…10

Продолжение табл. 1.1.

Тип трелевочной машины	M_s	1…20
Тип сучкорезной машины	M_c	1…20
Тип погрузчика	M_l	1…10
Тип технологического комплекта оборудования заготовки сортиментов	M_t	1…3
Способ раскряжевки	S_r	1…15
Метод раскряжевки	M_r	1…3
Калькуляционная статья (подстатья) расхода	u	1…20
Статья выпуска продукции (распределение услуг)	u'	1…20
Объем заготовленного материала, $м^3$	Q	0…∞
Количество заготовленного материала, шт	n	0…∞
Количество произведенного изделия (сортимента)	V	0…∞
Планово-расчетная цена за единицу материала, руб	c	0…∞
Планово-расчетная цена за единицу готового изделия, руб	r	0…∞
Абсолютная величина отклонения от нормы	±D	0…∞
Причина отклонения от нормы	w_D	1…20
Виновник отклонения от нормы	r_D	1…10

Название сортимента (пиломатериала)	M	1…5
Технические характеристики объекта:		
древесная порода	p	1…7
назначение (специальное)	α	1…17
способ распиловки	γ	1…3
Сорт	β	1…15
длина, мм	l	0…30
толщина, мм	a	0…10
ширина, мм	b	0…30
диаметр, мм	d	0…25

Продолжение табл. 1.1.

категория влажности	λ	1…3
Номер пакета	s	1…50
Продолжительность простоя оборудования, ч	$t_{пр}$	1…70
Причина простоя оборудования	$П_{ПР}$	1…20
Категория качества лесоматериалов	g	1…10

1.2.1. Механизированная и машинная валка деревьев

При осуществлении механизированной и машинной валки деревьев в журналах учета фиксируются учетные сообщения следующего вида:

ПП;I;e;DD;MM;YYYY;Num;F;R;T_t;S_f;S_p;M_v;Q;n

При этом наиболее частыми запросами к БД будут являться запросы по следующим полям:

-[ПП];[I];e;DD;MM;YYYY;Q;n

-ПП;I;e;DD;MM;YYYY;F;R;Q;n

-[ПП];[I];[e];DD;MM;YYYY;[T_t];S_f;S_p;M_v

-[DD];[MM];[YYYY];M_v;Q;n

-[DD];[MM];[YYYY];F;R;Q;n

-[DD];[MM];[YYYY];T_t;Q;n

-ПП;I;e;DD;MM;YYYY;F;R;T_t;M_v;g

Кроме того, к рассматриваемой БД будут выполняться статистические запросы на подсчет количества записей по комбинациям полей:

-F;MM;YYYY

-R;MM;YYYY

-S_f;MM;YYYY

-S_p;MM;YYYY

-M_v;MM;YYYY.

1.2.2. Технологические процессы трелевки древесины

Технология и организация трелевки древесины изложена в [127-129].

Исходными данными для ввода учетных сообщений о количестве перемещенных хлыстов в БД, являются маршрутные листы, заполняемые исполнителями последовательных операций трелевки древесины. Данные сообщения имеют следующую постоянную структуру:

ПП;I;e;DD;MM;YYYY;Num;p;S_t;M_s;Q;n

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

−e;[p;][Q;][S_t;][M_s;]DD;MM;YYYY;

−ПП;[p;][Q;][S_t;][M_s;]DD;MM;YYYY;

−I;[p;][Q;][S_t;][M_s;]DD;MM;YYYY;

−I;[ПП;][e;]DD;MM;YYYY.

1.2.3. Очистка деревьев от сучьев

Технология и организация очистки деревьев от сучков изложена в [123].

Работа по очистке деревьев характеризуется количеством обработанных хлыстов данной древесной породы по бригадам с использованием определенных типов сучкорезных машин, с учетом группы лесов и категории

рубки. Согласно записям в журнале учета формируют ежесменное учетное сообщение вида:

ПП;I;e;DD;MM;YYYY;Num;p;F;R;T_t;M_c;Q;n

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

– e;[p;]F;R;T_t;M_c;Q;DD;MM;YYYY;

– ПП;[p;] F;R;T_t;M_c;Q;DD;MM;YYYY;

– p;F;R;Q;DD;MM;YYYY;

– F;R;Q;DD;MM;YYYY.

Также к рассматриваемой таблице БД будут выполняться статистические запросы на подсчет количества записей, например, по комбинациям полей e;MM;YYYY и Q;MM;YYYY.

1.2.4. Погрузка древесины на верхних складах

Технология и организация погрузки древесины подробно изложена в [124-126].

Фиксируемые учетные сообщения, в данном случае, имеют вид:

ПП;I;e;DD;MM;YYYY;Num;p;F;R;M_l;Q;n

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

– e;[p;][F;][R;]Q;DD;MM;YYYY;

– ПП;[p;][F;][R;]Q;DD;MM;YYYY;

– [p;][F;][R;]Q;[ПП;][e;]DD;MM;YYYY.

1.2.5. Заготовка сортиментов на лесосеке

Технология и организация раскряжевки хлыстов подробно изложена в [118-122]. Основным первичным документом, отражающим работу бригад, является сменный рапорт.

Обработка данных сменного рапорта позволяет получать следующие документы:

−ведомость заготовленных сортиментов;

−регистров, характеризующих использование каждой бригады в связи с применяемой технологией раскряжевки;

−ведомость выполнения плана заготовки сортиментов;

−ведомость объемов сортиментов по группам древесных пород, назначению и диаметрам за каждые сутки отчетного месяца в штуках и объему по каждой позиции диаметров;

−ведомость объемов произведенных работ по сменам (за каждую дату), бригадам, группам древесных пород и назначению сортиментов в штуках, погонных и кубических метрах с распределением объема по видам сортиментов и фиксацией кубомассы; в данной ведомости могут фигурировать и ежемесячные итоговые сведения о простоях оборудования;

−другие ведомости и обобщающие группировки данных.

Формирование сменного рапорта происходит путем фиксирования в БД КИС предприятия ежесменных учетных сообщений следующего вида:

ПП;I;e;DD;MM;YYYY;Num;p;β;l;d;$t_{ПР}$;$П_{ПР}$;n;Q;c;S_r;M_r;M_t

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

−e;[p;][β;][l;][d;]DD;MM;YYYY;

−ПП;[p;][β;][l;][d;]DD;MM;YYYY;

−S_r;[p;][β;][l;][d;]DD;MM;YYYY;

−M_r;[p;][β;][l;][d;]DD;MM;YYYY;

−M_t;[p;][β;][l;][d;]DD;MM;YYYY;

−p;[β;]DD;MM;YYYY;

−β;DD;MM;YYYY;

−l;[d;]DD;MM;YYYY;

−d;DD;MM;YYYY;

−$t_{ПР}$;$П_{ПР}$;ПП;[e;][I;]DD;MM;YYYY.

Также к рассматриваемой таблице БД будут выполняться статистические запросы на подсчет количества записей, например, по комбинациям полей:

– e;MM;YYYY;

– ПП;MM;YYYY;

– $П_{ПР}$;MM;YYYY.

В целях обеспечения учетного контроля процесса раскряжевки хлыстов на сортименты, по данным текущей регистрации создаются ежесменные учетные сообщения с информацией о полученных сортиментах по признакам древесной породы, назначения, характера обработки, сорта, длины, толщины и ширины пиломатериала (или с меньшим набором признаков). Чаще всего учетное сообщение строится по схеме:

ПП;I;e;DD;MM;YYYY;Num;M;p;α;γ;β;l;V;r;P

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

– ПП;[M;][p;][α;][γ;][β;][l;]DD;MM;YYYY;

– e;[M;][p;][α;][γ;][β;][l;]DD;MM;YYYY;

– I;[M;][p;][α;][γ;][β;][l;]DD;MM;YYYY;

– M;[p;][α;][γ;][β;][l;]DD;MM;YYYY.

Также к рассматриваемой таблице БД будут выполняться статистические запросы на подсчет количества записей, например, по комбинациям полей ПП;MM;YYYY и e;MM;YYYY.

1.2.6. Лесопильное производство

Технология и организация лесопильного производства подробно изложена в [70-74].

Первичные данные о распиленном пиловочнике по всем товарным признакам фиксируют в сменных рапортах. Посменная регистрация раскроя пиловочника и выпуска пиломатериалов на отдельных потоках лесопильного цеха позволяет оперативно получать как натуральные показатели

качественного, спецификационного и количественного выходов, так и синтетический показатель ценностного выхода по каждой бригаде. Поскольку ценностный выход пиломатериалов наиболее полно характеризует усилия бригады в достижении оптимального уровня использования пиловочного сырья, раздельный (по потокам) учет пиломатериалов на сортировочной площадке по древесным породам, размерам, сортам и назначению абсолютно необходим. Последующая группировка выпущенных изделий должна обеспечить побригадную оценку выпуска в оптовых ценах на каждом потоке за любой отчетный период.

Сменный рапорт о распиловке леса на поточной линии - специфический документ, служащий одновременно для фиксации производственного потребления пиловочника и для отражения работы лесопильных рам по его распилу. Поэтому в рапорт записывают все основные признаки, характеризующие распиленный пиловочник: сортимент (назначение, порода и сорт), размеры (длину и диаметр) и количество (в штуках и кубометрах) по каждому сочетанию длины и диаметра пиловочника.

Последующий свод исходной информации, содержащейся в сменных рапортах о распиловке леса, приводит к получению различных обобщающих документов: во-первых, ведомостей распиленного сырья, во-вторых, регистров, характеризующих использование каждой лесопильной линии в связи с применяемой технологией раскроя древесины.

Для оперативного управления раскроем пиловочного сырья на деревообрабатывающих предприятиях используется обычно следующий перечень сводных документов:

– ведомость выполнения плана раскроя сырья, в которой приводятся ежесуточные сведения о количестве распиленного пиловочника в штуках данных диаметров по номерам поставов, принятых в плане раскроя;

- ведомость объемов распиленного пиловочника по группам древесных пород, назначению и диаметрам бревен за каждые сутки отчетного месяца в штуках и объему по каждой позиции диаметров;
- ведомость объемов распиленного пиловочника по сменам (за каждую дату), рамным потокам (бригадам), группам древесных пород и назначению пиловочника в штуках, погонных и кубических метрах с распределением объема по видам распиловки (вразвал, с брусовкой) и фиксацией кубомассы; в данной ведомости могут фигурировать и ежемесячные итоговые сведения о простоях лесопильных рам;
- другие ведомости и обобщающие группировки данных о распиленном лесе.

Для правильного определения стоимости пиловочного сырья, раскроенного хозрасчетной бригадой за смену, декаду, месяц, необходимо создавать ежесменные учетные сообщения следующего вида:

$$ПП;I;e;DD;MM;YYYY;Num;p;\beta;l;d;t_{ПР};П_{ПР};n;Q;c;P \quad (1.1)$$

Расчетное количество записей этого типа, накапливаемое в БД за год, определяется по схеме:

[количество участков в цехе]*[количество смен в сутки]*[количество рабочих дней в году]*[количество древесных пород пиловочника]*[количество значений назначения]*[количество сортов]*[количество значений длины]*[количество значений диаметра]

и составляет:

- максимальное - 1735020000;
- среднее - 622080.

По аналогичной схеме определяется расчетное количество записей всех типов записей, рассматриваемых далее.

Размер файла БД при среднем кол-ве записей будет равен 42*622080=26127360 байт, где 42 – это размер одной записи данного типа в байтах.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- e;[p;][β;][l;][d;]DD;MM;YYYY;
- ПП;[p;][β;][l;][d;]DD;MM;YYYY;
- p;[β;]DD;MM;YYYY;
- β;DD;MM;YYYY;
- l;[d;]DD;MM;YYYY;
- d;DD;MM;YYYY;
- $t_{ПР}$;$П_{ПР}$;ПП;[e;][I;]DD;MM;YYYY.

Также к рассматриваемой таблице БД будут выполняться статистические запросы на подсчет количества записей, например, по комбинациям полей:

- e;MM;YYYY;
- ПП;MM;YYYY;
- $П_{ПР}$;MM;YYYY.

В лесопильном производстве под учетным контролем находится далее процесс основной сортировки досок-полуфабрикатов, полученных из раскроенного пиловочника. На этой производственной стадии по данным текущей регистрации создаются ежесменные учетные сообщения с информацией о выработанных пиломатериалах по признакам древесной породы, назначения, характера обработки, сорта, длины, толщины и ширины пиломатериала (или с меньшим набором признаков). Чаще всего учетное сообщение строится по схеме

ПП;I;e;DD;MM;YYYY;Num;M;p;α;γ;β;l;a;b;V;r;P (1.2)

Параметры, содержащиеся в учетном сообщении (1.2), позволяют предварительно оценить доски-полуфабрикаты, произведенные бригадой или сменой цеха, в оптовых ценах.

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное - 27760320000;

– среднее – 2799360.

Размер файла БД при среднем количестве записей будет равен 134369280 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

– ПП;[M;][p;][α;][γ;][β;][l;][a;][b;]DD;MM;YYYY;
– e;[M;][p;][α;][γ;][β;][l;][a;][b;]DD;MM;YYYY;
– I;[M;][p;][α;][γ;][β;][l;][a;][b;]DD;MM;YYYY;
– M;[p;][α;][γ;][β;][l;][a;][b;]DD;MM;YYYY.

Также к рассматриваемой таблице БД будут выполняться статистические запросы на подсчет количества записей, например, по комбинациям полей ПП;MM;YYYY и e;MM;YYYY.

Последующая автоматизированная обработка сообщений (1.1) и (1.2) обеспечивает проведение анализа эффективности применяемой технологии лесопиления.

Специальные учетные сообщения полезно создавать на участке формирования технологических пакетов, на торцовочно-маркировочной установке, на участке сортировки досок по длинам и укладки их в транспортные пакеты, а также и на других технологических участках выработки пиломатериалов.

Рассмотрим практику фиксации данных о двух вспомогательных процессах лесопильного производства - окорке пиловочного сырья и сушке пиломатериалов.

Подачу бревен (в штуках и кубических метрах) для окорки регистрируют в сменных рапортах за смену, отработанную бригадой, по древесным породам, длинам и диаметрам бревна. На основе данных в сменных рапортах формируют учетные сообщения вида

$$ПП;I;e;DD;MM;YYYY;Num;p;l;d;t_{ПР};П_{ПР};n;Q \qquad (1.3)$$

Путем осуществления запросов к таблице записей данного вида будут получены ведомости распределения количества окоренных бревен определенных пород за сутки, декаду, месяц по длинам и диаметрам (штуки, кубические метры), а также другие группировки, необходимые для текущего контроля за уровнем выполнения заданий на окорку и для начисления заработной платы членам бригад, обслуживающих окорочные агрегаты.

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 22680000;
- среднее – 51840.

Размер файла БД при среднем количестве записей будет равен 1555200 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- ПП;[p;][l;][d;]DD;MM;YYYY;
- e;[p;][l;][d;]DD;MM;YYYY;
- l;[d;]DD;MM;YYYY;
- d;DD;MM;YYYY;
- $t_{ПР}$;$П_{ПР}$;ПП;[e;][I;]DD;MM;YYYY.

Также к рассматриваемой таблице БД будут выполняться статистические запросы на подсчет количества записей, например, по комбинациям полей ПП;MM;YYYY и e;MM;YYYY.

Данные, фиксируемые в регистрах первичного учета сушки пиломатериалов, используют для формирования сообщений вида

I;e;DD;MM;YYYY;Num;S;s;M;p;l;a;b;n;g;J;$t_Ф$;$t_{ПР}$;$П_{ПР}$;Q (1.4)

В сообщениях (1.4) содержатся основные сведения, необходимые для технологической характеристики сушильных штабелей, процесса камерной сушки, объема высушенных пиломатериалов, использования сушильной камеры, а также для расчетов с обслуживающим персоналом. Автоматизация

обработки информации, включаемой в эти сообщения, заметно повышает уровень оперативного анализа работы сушильного цеха за любые отрезки времени.

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 181440000;
- среднее – 1166400.

Размер файла БД при среднем количестве записей будет равен 47822400 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- s;[M;][p;][l;][a;][b;][g;][J;]DD;MM;YYYY;
- S;[M;][p;][l;][a;][b;][g;][J;]DD;MM;YYYY;
- e;[M;][S;][p;][l;][a;][b;]DD;MM;YYYY;
- p;[M;][l;][a;][b;][g;]DD;MM;YYYY;
- a;[M;][l;][b;][g;]DD;MM;YYYY;
- b;[M;][l;][g;]DD;MM;YYYY;
- $t_{ПР}$;$П_{ПР}$;[S;]DD;MM;YYYY.

КИС лесопильного производства должна путем обработки первичных данных (записей БД) предоставлять следующую обобщенную информацию:

- натуральные и ценностные показатели переработки сырьевых материалов;
- показатели, характеризующие использование оборудования в связи с применяемыми технологиями производства пиломатериалов;
- информацию о выработанных пиломатериалах по признакам древесной породы, назначения, характера обработки, сорта и размера;

1.2.7. Фанерное производство

Технология и организация фанерного производства изложена в [78].

Рассмотрим особенности регистрации выработки фанеры по процессам раскроя чураков на шпон, сушки шпона, склеивания фанеры.

Работа лущильного отделения характеризуется количеством израсходованного фанерного сырья данной древесной породы и налущенного сырого полноформатного шпона ($V_{СПШ}$), деловых кусков ($V_{СКШ}$) и малоформатного шпона из карандашей ($V_{СМШ}$). Согласно записям в журнале учета формируют ежесменное учетное сообщение вида

ПП;I;e;DD;MM;YYYY;Num;p;β;l;a;b;Q;$V_{СПШ}$;$V_{СКШ}$;$V_{СМШ}$ (1.5)

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 45360;
- среднее – 12960.

Размер файла БД при среднем количестве записей будет равен 492480 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- e;[p;][β;][l;][a;][b;]DD;MM;YYYY;
- ПП;[p;][β;][l;][a;][b;]DD;MM;YYYY;
- p;[β;][l;][a;][b;]DD;MM;YYYY;
- β;[p;][l;][a;][b;]DD;MM;YYYY.

В сушильном отделении по каждой смене фиксируют количество просушенного шпона ($V_{СШ}$) и переданного сухого шпона (полного формата ($V_{СПШ}$), кусков ($V_{СКШ}$), малых форматов ($V_{СМШ}$)). На основе этих записей создаются учетные сообщения вида

S;I;e;DD;MM;YYYY;Num;p;β;$V_{СШ}$;$V_{СПШ}$;$V_{СКШ}$;$V_{СМШ}$ (1.6)

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 120960;
- среднее – 12960.

Размер файла БД при среднем количестве записей будет равен 505440 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- e;[p;][β;]DD;MM;YYYY;
- S;[p;][β;]DD;MM;YYYY;
- p;[β;]DD;MM;YYYY;
- β;DD;MM;YYYY.

Клеильному отделению принадлежит наибольшее количество фиксируемых показателей фанерного производства. В учетных сообщениях, формируемых на этом участке согласно журналам учета клейки фанеры, фигурируют данные о каждой запрессовке, произведенной клеильными прессами отделения. Структура сообщения о клейке имеет вид

$$I;e;A;DD;MM;YYYY;Num;Ф;p;\beta;E;Ш;l;a;b;{}^{0}C_{ПЛИТ};P_{АТМ};t_{Ф};V \qquad (1.7)$$

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 680400000;
- среднее – 2099520.

Размер файла БД при среднем количестве записей будет равен 100776960 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- e;[A;][Ф;][E;][Ш;][p;][β;][l;][a;][b;]DD;MM;YYYY;
- A;[Ф;][E;][Ш;][p;][β;][l;][a;][b;]DD;MM;YYYY;
- Ф;[E;][Ш;][p;][β;][l;][a;][b;]DD;MM;YYYY;
- p;[E;][Ш;][β;][l;][a;][b;]DD;MM;YYYY;
- β;[E;][Ш;][l;][a;][b;]DD;MM;YYYY;
- l;a;b;DD;MM;YYYY.

Также к рассматриваемой таблице БД будут выполняться статистические запросы на подсчет количества записей, например, по комбинациям полей e;MM;YYYY и A;MM;YYYY.

1.2.8. Производство древесных плит

Технология и организация производства древесных плит подробно изложена в [75-77].

Формирование учетных сообщений об изготовлении древесностружечных и древесноволокнистых плит основано на ежесменной регистрации данных об этих процессах в журналах учета.

При производстве древесноволокнистых плит регистрируют следующие показатели:

1. Остатки технологической щепы в бункерах ($Q_{БУН}^{ОСТ}$) и древесной массы в бассейнах ($Q_{БАС}^{ОСТ}$), расход щепы на отлив плит ($Q^{РАС}$), выработку щепы ($Q^{ВЫР}$).

Данные учетные сообщения имеют вид

ПП;I;e;DD;MM;YYYY;Num;$Q_{БУН}^{ОСТ}$;$Q_{БАС}^{ОСТ}$;$Q^{РАС}$;$Q^{ВЫР}$ (1.8)

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 6480;
- среднее – 6480.

Размер файла БД при среднем количестве записей будет равен 239760 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- e;DD;MM;YYYY;
- ПП;DD;MM;YYYY.

2. Количество отпрессованных (отлитых) ($V_{ОТЛ}$), обрезанных и сданных на склад годных ($V_{ОБ\ ГОД}$) и забракованных плит ($V_{ОБ\ БР}$) (по каждой их разновидности).

Данные учетные сообщения имеют вид

ПП;I;e;DD;MM;YYYY;Num;B;$V_{ОТЛ}$;$V_{ОБ\ ГОД}$;$V_{ОБ\ БР}$ (1.9)

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 32400;
- среднее – 12960.

Размер файла БД при среднем количестве записей будет равен 349920 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- e;[B;]DD;MM;YYYY;
- ПП;[B;]DD;MM;YYYY;
- B;[ПП;][e;]DD;MM;YYYY.

3. Данные о нормированном ($Q_{НОРМ}$) и фактическом ($Q_{ФАКТ}$) расходах химикатов (по видам) на выработку плит (по их разновидностям), о числе варок и количестве сваренной эмульсии (V), о фактическом расходе данного химиката на выпуск изделий и об отклонениях от норм расхода.

Данные учетные сообщения имеют вид

ПП;I;e;DD;MM;YYYY;Num;B;N;V;$Q_{НОРМ}$;$Q_{ФАКТ}$;±D (1.10)

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 32400;
- среднее – 12960.

Размер файла БД при среднем количестве записей будет равен 518400 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- e;[B;]DD;MM;YYYY;
- ПП;[B;]DD;MM;YYYY;
- B;[ПП;][e;]DD;MM;YYYY.

В производстве древесностружечных плит формируют аналогичные учетные сообщения.

1.2.9. Мебельное производство

Технология и организация мебельного производства изложена в [79-81].

Учет процесса изготовления мебели сопровождается составлением большого количества учетных сообщений, фиксируемых в БД .

Рассмотрим сообщения, несущие информацию об изготовлении деталей из ДСП при производстве корпусной мебели. Исходными данными для ввода записей в БД являются маршрутные листы, заполняемые исполнителями последовательных операций изготовления деталей. Данные сообщения имеют следующую постоянную структуру:

$$ПП;I;e;DD;MM;YYYY;Num;B;B_{ДЕТ};l;a;b;V_1;\ldots;V_n \qquad (1.11)$$

где n – количество этапов обработки деталей.

Расчетное количество записей этого типа, накапливаемое в БД за год, составляет:

- максимальное – 5832000;
- среднее – 259200.

Размер файла БД при среднем количестве записей будет равен 8035200 байт.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

- $e;[B;][B_{ДЕТ};][V_1;][\ldots][V_n;]DD;MM;YYYY;$
- $ПП;[e;][B;][B_{ДЕТ};][V_1;][\ldots][V_n;]DD;MM;YYYY;$
- $B_{ДЕТ};[B;][V_1;][\ldots][V_n;][ПП;][e;]DD;MM;YYYY;$
- $B;[ПП;][e;]DD;MM;YYYY.$

Сообщения о плановых величинах по различным стадиям производства имеют сходную структуру и хранятся в отдельной таблице плановых показателей.

Путем осуществления запросов к таблице учетных сообщений и плановых величин в рамках АСУ мебельного производства будут получены следующие расчетные данные по различным стадиям производства:

- трудовые издержки производственных рабочих по нормам и по факту, в трудочасах и денежном выражении;
- показатели выполнения плана производства;
- расходы на содержание и эксплуатацию оборудования по сметным ставкам, установленным на изделие того или иного артикула, и фактические.

1.2.10. Вспомогательные подразделения

Отпуск однородной продукции вспомогательных подразделений внутризаводским потребителям по планово-расчетной оценке, отражается учетными сообщениями типа

ξ;DD;MM;YYYY;Num;B;ξ';V;r;P (1.12)

где ξ'- структурное подразделение - потребитель услуг.

Наиболее частыми запросами к данной таблице БД будут являться запросы по следующим полям (комбинациям полей):

−ξ;[B;][ξ';]DD;MM;YYYY;

−B;[ξ';]DD;MM;YYYY;

−ξ';[B;]DD;MM;YYYY.

1.3. Методы доступа к БД информационной системы управления лесопромышленного предприятия

В работах [130, 132] был проведен анализ существующих в настоящее время методов доступа к данным и файлов различных структур данных. На основе проведенного анализа метод индексирования (метод плотных вторичных комбинированных индексов) был представлен в качестве наиболее эффективного метода организации доступа к данным БД КИС лесопромышленного предприятия. Основные показатели эффективности применения данного метода отражены в [130].

В [132] автором предлагается и обосновывается повышение эффективности метода индексирования за счет модернизации алгоритма построения индекса [131] с учетом специфики информационной модели БД лесопромышленного предприятия и за счет сжатого хранения указателей на странице с записями. Предлагаемый метод индексирования получил название усовершенствованного метода комбинированных индексов. Автором доказывается, что усовершенствованный метод комбинированных индексов при построении информационной модели БД лесопромышленного предприятия имеет следующие преимущества перед стандартным методом индексирования:

−Возможность применения при запросах по любой комбинации индексированных полей.

−Меньший размер индексного файла по сравнению с размером стандартного индекса вследствие сжатого хранения комбинаций значений индексируемых полей и указателей в файле индекса.

−Возможность осуществления статистических запросов (подсчета количества записей, удовлетворяющих определенному поисковому критерию) без запроса к таблице данных, что позволяет значительно сократить время запроса.

−Возможность удаления индексированных полей из таблицы данных без потери информации. Это приведет к снижению размера БД.

−В случае осуществления только статистических запросов к БД таблица данных может быть полностью удалена, что значительно сократит размер БД.

Таким образом, в качестве основного метода доступа к файлам БД КИС лесопромышленного предприятия предлагается использовать усовершенствованный метод комбинированных индексов.

1.4. Выводы

По результатам анализа структуры КИС и размера файлов БД можно сделать следующие выводы о свойствах информационной модели КИС лесопромышленного предприятия:

1. Большой размер файлов БД КИС, объемы и широкий перечень элементов записей (полей) требуют детального анализа структуры хранения данных в БД КИС и оптимизации структуры файлов БД для обеспечения производительности системы.

2. Большинство запросов к БД КИС осуществляется по комбинациям полей (4-6 и более полей).

3. Статистические запросы к БД по подсчету количества записей, удовлетворяющих критериям поиска, также осуществляются по комбинациям из нескольких полей.

4. Наиболее эффективным методом доступа к файлам БД КИС лесопромышленного предприятия является усовершенствованный метод комбинированных индексов.

5. В целях снижения затрат, при создании КИС лесопромышленного предприятия, следует стремиться к использованию такого модульного состава ИО КИС, который позволит предприятию самостоятельно осуществлять настройки КИС, и адаптировать систему под конкретные технологические, отраслевые и иные особенности каждого лесопромышленного предприятия.

Таким образом, будем считать сделанные выводы основными критериями для выбора средства реализации КИС лесопромышленного предприятия в рамках главы 2.

2. СПЕЦИФИКА СРЕДЫ РЕАЛИЗАЦИИ ИНФОРМАЦИОННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

При выборе средств реализации КИС лесопромышленного предприятия на основании выводов, сделанных в главе 1, следует руководствоваться следующими критериями:

1. Уровень СУБД (ядро системы):
   - Структура данных СУБД должна предусматривать возможность проектирования многоуровневых структур данных, что является необходимым для учета специфики информационной модели БД КИС лесопромышленного предприятия.
   - Структура хранения данных должна обеспечивать возможностью использования инвертированных списков и хранения инвертированных структур.
   - Методы доступа к данным СУБД должны обладать мощным инструментарием поиска, чтения и селекции данных; производительными методами осуществления запросов к данным по нескольким критериям (комбинациям из нескольких полей), в том числе, к инвертированным спискам и инвертированным структурам.
   - Целостность данных СУБД.
2. Модульный уровень системы (программы взаимодействия с ядром системы, основные расчетные программы, пользовательские приложения, отчетные формы):
   - Инструментарий для создания экономико-математических моделей и расчетов, а также, интерпретации данных в рамках программных модулей КИС.
   - Наличие эффективных средств взаимодействия приложений системы с БД КИС, механизмов передачи данных из БД в расчетные модули и обратно.

– Возможность осуществления настройки системы и адаптации КИС к конкретным особенностям (технологическим, организационным и пр.) лесопромышленного предприятия.

– Возможность представления средств управления и настройки системы, а также, отчетных форм в графической, понятной пользователю, форме.

3. Уровень аппаратного обеспечения и корпоративной сети:

– Возможность реализации КИС на различных аппаратных платформах.

– Возможность функционирования КИС в условиях распределенной среды.

4. Оптимальная стоимость ПО среды проектирования и эксплуатации.

Был проведен анализ существующих СУБД и редакторов приложений на соответствие вышеуказанным критериям. На основе анализа были сделаны выводы, что наиболее полно отражающей свойства информационной модели лесопромышленного предприятия средой построения КИС является СУБД ADABAS и редактор Natural.

Далее приведены характеристики среды выбранной среды проектирования и эксплуатации КИС лесопромышленного предприятия.

2.1. Специфика СУБД ADABAS

Как было отмечено в главе 1, при построении КИС лесопромышленного предприятия неизбежно хранение и обработка больших объемов данных. Кроме того, для моделирования сложной структуры информационной модели лесопромышленного предприятия необходимо учесть ряд требований к СУБД.

Современное состояние науки о базах данных, в том числе особенности архитектуры и методы доступа к данным, рассмотрено в [45, 133-149].

Производительность КИС лесопромышленного предприятия во многом зависит от средств и возможностей СУБД, при помощи средств которой строится ядро системы.

1. Модель и структура данных СУБД.
2. Структура хранения данных СУБД.
3. Методы доступа к данным СУБД.
4. Ограничения целостности.

2.1.1. Модель и структура данных

Подробно модели и структуры данных различных СУБД изложены в работах [45, 150-157]. База данных ADABAS определяется как совокупность взаимосвязанных файлов и ассоциаций записей файлов.

Файл представляет собой именованную совокупность записей, имеющих одинаковую структуру, компонентами которой являются:

- атрибут и множественный атрибут;
- простая, составная и повторяющаяся группа.

Каждой записи файла назначается системный ключ, представляющий собой число в диапазоне от 1 до 16 * 106, которое однозначно идентифицирует экземпляр записи и определяется как внутрисистемный номер. Связи между файлами (1:1; 1:M; M:N) являются непоименованными парными связями и предназначены для моделирования сетевых и иерархических отношений между объектами предметной области, а также групповых отношений в объектах иерархической структуры.

Ассоциация записей файлов (далее ассоциация) представляет собой совокупность записей файла, обладающих общим свойством. Ассоциации организуются в БД в виде списков внутрисистемных номеров (ВСН) записей файлов. Если ассоциация образуется на основе равенства значений некоторого атрибута, список ВСН записей, входящих в ассоциацию, называется инвертированным списком, а сам атрибут — поисковым. В том случае, когда ассоциация образуется на основе равенства значений атрибута записей одного файла, значению атрибута некоторой записи другого файла, соответствующий список ВСН называется списком связи, а указанные атрибуты файлов - атрибутами связи.

Использование ассоциаций реализует, по существу, разбиение записей файлов на классы, поддерживаемое динамически в процессе модификации БД. При этом ассоциации в виде инвертированных списков обеспечивают возможность ускоренной селекции записей файлов по их содержимому, а ассоциации в виде списков связь - возможность поддержания связей между файлами и возможность селекции записей с применением этих связей.

Атрибут является наименьшей именованной единицей данных, каждый экземпляр которой в записи файла представляется одним или несколькими значениями.

Для атрибута в схеме файла задаются его короткое двухсимвольное имя, тип и максимальная длина значения.

Тип значения атрибута определяет стандартный вид представления значения этого атрибута в прикладной программе. К числу допустимых типов значений относится символьный, упакованный и распакованный десятичный, битовый и двоичный.

Атрибут в схеме файла может быть описан как уникальный, при этом его значения однозначно определяют экземпляры записей в файле БД и могут быть использованы в качестве ключей записей.

Группа представляет собой именованную совокупность атрибутов и, возможно, других групп.

Простая группа состоит только из атрибутов, составная группа может содержать как атрибуты, так и простые и составные группы.

Периодической называется группа, которая может иметь в записи несколько экземпляров. Периодические группы в структуре файла не должны быть вложенными, т. е. не должны входить в состав других групп. Простые и составные группы могут быть иерархически подчинены одна другой и могут входить в состав повторяющихся групп, образуя древовидную структуру.

Периодическая группа является средством реализации связи типа 1:M в пределах записи. Пример использования периодической группы в составе структуры файла СОТРУДНИКИ представлен на рис. 2.1.

Рис. 2.1. Записи файла СОТРУДНИКИ с периодической группой СЛУЖЕБНЫЕ ДАННЫЕ

Согласно рис. 2.1. у работника одного предприятия с продвижением по службе могут быть зафиксированы несколько должностей, с указанием размера оклада и даты назначения на соответствующий пост.

Ассоциации записей файла в БД организуются в виде инвертированных списков по значениям поисковых атрибутов и в виде списков связи - по значениям атрибутов связи.

Ассоциации, организованные по значениям производных атрибутов, расширяют возможности динамической классификации записей файла, позволяя моделировать классификацию объектов предметной области не только по их свойствам, но и по комбинации свойств, что увеличивает производительность системы при осуществлении поиска по нескольким критериям.

Ассоциации записей файлов БД, реализованные посредством инвертированных списков и списков связи, автоматически создаются и корректируются при вводе, корректировке и удалении записей файлов, содержащих значения поисковых атрибутов, атрибутов связи и значения атрибутов, входящих в состав значений производных атрибутов.

Пример организации инвертированных списков иллюстрируется рис. 2.2. Каждая запись представлена с ее внутрисистемным номером, который не входит в состав структуры записи. Поисковыми в схеме файла КАДРЫ

являются атрибуты ГОД-РОЖДЕНИЯ и ПОЛ с кодами имен соответственно GR и PL. Файл КАДРЫ первоначально содержит восемь записей. По значениям поискового атрибута ГОД-РОЖДЕНИЯ для представленных записей организовано три инвертированных списка, по значениям поискового атрибута ПОЛ - два.

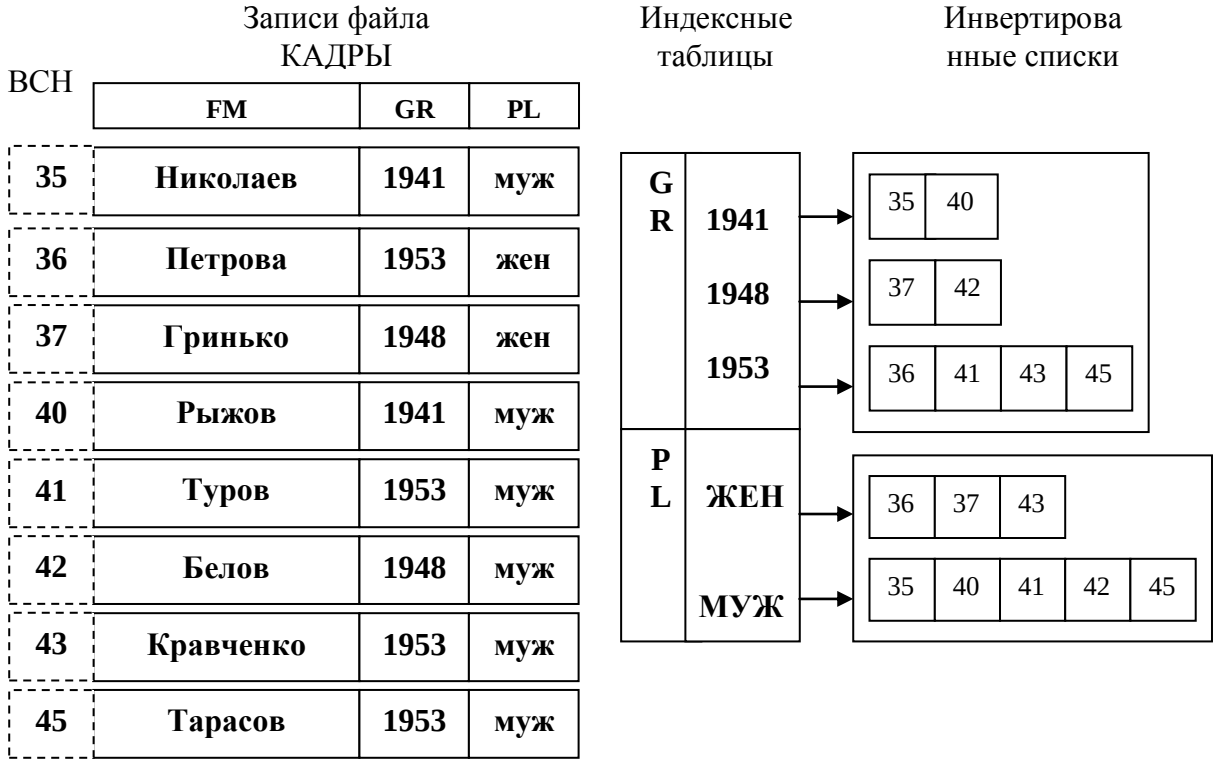


Рис. 2.2. Записи файла и инвертированные списки до модификации файла

Рассмотрим далее организацию связей между записями файлов БД. Между любыми двумя файлами БД возможна организация одного типа связи, причем каждый файл может быть связан с несколькими другими файлами.

Запись любого файлов объявляется связанной с одной или несколькими записями второго файла, если атрибут связи в каждой из этих записей имеет значение, совпадающее со значением атрибута связи в записи первого файла. Атрибуты связи в каждом из связываемых файлов должны быть поисковыми (объявленными дескрипторами), поскольку формирование списков связи

осуществляется с помощью инвертированных списков, которые до связывания должны быть организованы по каждому из этих атрибутов. Для организации связи типа один-к-одному в качестве атрибутов связи в каждом из файлов должны быть объявлены уникальные атрибуты. Связь типа один-ко-многим и типа многие-к-одному будет организована, если только один из атрибутов связи будет уникальным, а второй - простым атрибутом.

2.1.2. Структура хранения данных

Существующие структуры хранения данных рассмотрены в [45, 158-162].

База данных ADABAS размещается на устройствах прямого доступа. Записи БД запоминаются в блоках устройств прямого доступа. Размер блока выбирается с учетом условия эффективного размещения целого числа блоков на дорожке используемых типов устройств прямого доступа.

Размещение записей в блоке обеспечивает более эффективное использование внешней памяти.

Структурными элементами базы данных ADABAS на внутреннем уровне являются: накопитель, ассоциатор, рабочий набор и вспомогательные наборы данных. Накопитель, ассоциатор и рабочий набор размещаются в наборах данных ОС ЕС, которые имеют прямую организацию и могут быть многотомными.

Структура накопителя. Накопитель, который занимает не более пяти наборов данных, предназначен для хранения записей файла БД. Каждому файлу выделяется до пяти экстентов, размещение которых отмечается в таблице размещения файла (ТРФ), находящейся в ассоциаторе.

Размер блока зависит от типа устройства внешней памяти. Блок набора данных накопителя состоит из поля длины, содержащего общую длину занятой части блока, и записи файла. Размер записи не должен превышать размера блока. В блоке может находиться несколько записей.

Запись содержит поле длины и поле BCH (ISN), за которыми следуют значения атрибутов. BCH является уникальным ключом записи. Физический адрес блока, в котором находится запись с данными BCH, определяется посредством преобразователя адреса, размещенного в ассоциаторе.

Значения множественного атрибута хранятся как последовательность значений с указателем длины и предшествующим однобайтовым счетчиком числа экземпляров. Периодическая группа начинается с однобайтового счетчика, указывающего самый большой номер из хранимых в группе экземпляров.

На рис. 2.3. представлена структура записи файла СТУДЕНТ для следующей карточки студента:

НОМЕР КАРТОЧКИ	115573
ФАМИЛИЯ	Ухов
ИМЯ-ОТЧЕСТВО	Петр Иванович
УЧЕБНАЯ ГРУППА	241
СЕССИЯ	1
ЭКЗАМЕНЫ	математика, физика, электротехника
ОЦЕНКИ	5, 4, 4
СЕССИЯ	2
ЭКЗАМЕНЫ	математика, физика, электротехника
ОТМЕТКИ	4, 4, 5

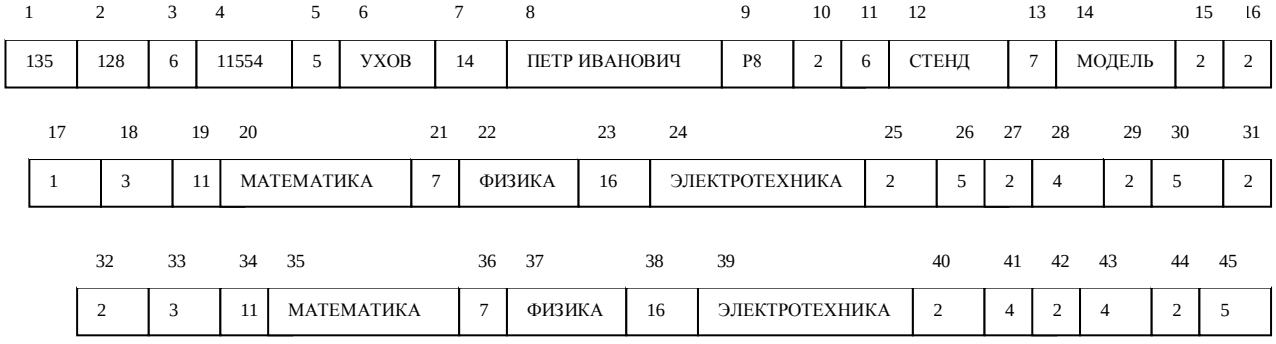


Рис. 2.3. Структура записи файла СТУДЕНТ

Поля записи (для удобства они пронумерованы) содержат:

1 - общую длину записи;

2 - BCH;

3, 5, 7, 11, 13, 2, 19, 21, 23, 25, 27, 29, 31, 34, 36, 38, 40, 42, 44 - длину значений атрибутов;

4, 6, 8, 9, 12, 14, 17, 20, 22, 24, 26, 28, 30, 32, 35, 37, 39, 41, 43, 45 - значения атрибутов;

10, 18, 33 — счетчики экземпляров множественного атрибута;

15 - счетчик экземпляров повторяющейся группы.

Структура ассоциатора. Ассоциатор содержит сведения о структуре данных концептуального и внутреннего уровней БД в виде таблиц, списков и т. д., помещенных в блоки, и служит для взаимного отображения этих структур и выполнения операций над ними.

Таблицы и списки ассоциатора формируются при создании и ведении БД с помощью словаря данных и модифицируются системой во время ее функционирования. Основная информация о размещении БД сосредоточена в таблице распределения памяти (ТРП).

ТРП занимает один блок ассоциатора и имеет постоянный адрес. Она содержит:

- имя и номер БД;
- максимальное число файлов, которые могут быть загружены в БД;
- число загруженных файлов;
- номера системных файлов;
- адреса экстентов, выделенных для ассоциатора, накопителя, рабочего набора;
- типы устройств, на которых они размещены, неиспользованные области памяти и т. д.

Данные о размещении файла в БД содержит таблице размещения файла (ТРФ). Каждому файлу соответствует своя ТРФ, которая занимает один блок ассоциатора. Ее местоположение в ассоциаторе определяется номером файла.

ТРФ составляется средствами АБД во время создания файла БД. В ней содержатся:

– имя файла;

– номер файла;

– имя атрибута рандомизированного доступа;

– загрузочная схема связей файлов;

– коэффициент заполнения блоков файла в накопителе;

– коэффициент заполнения блоков инвертированных списков в ассоциаторе;

– сведения о размещении файла в накопителе, инвертированных списков и их индексов в ассоциаторе, преобразователя ВСН записей в адрес блока файла, таблицы учета свободной памяти в блоках накопителя и т. д.

Описание логической структуры файла транслируется в таблицу определения данных (таблица FDT), которая представляет собой загрузочную форму схемы файла.

Таблица FDT для каждого файла имеет фиксированный адрес и занимает 4 блока ассоциатора, поля этой таблицы содержат признаки свойств каждого атрибута файла.

Ряд операций, инициируемых командами ЯМД, выполняется с помощью инвертированных списков, состоящих из ВСН записей файла.

Инвертированные списки хранятся в блоке набора данных ассоциатора совместно с заголовочной частью, состоящей из значения атрибута, которому соответствует список, длины этого значения и длины инвертированного списка. В одном блоке размещается несколько списков, но все они должны соответствовать значениям одного атрибута. Список отсортирован в порядке возрастания значений ВСН, поэтому первый ВСН оказывается младшим элементом списка.

Доступ к требуемому инвертированному списку осуществляется с помощью многоуровневого индекса, содержащего индекс значений и старшие индексы (рис. 2.4.).

Индекс значения состоит из записей, каждая из которых включает длину атрибута, его значение, первый BCH списка и адрес блока инвертированных списков. Значение атрибута то же, что и в первом инвертированном списке адресованного блока, а BCH является младшим в этом списке и указывает на то, что список не имеет продолжения. Таких записей в индексе значений будет столько, сколько блоков занимают инвертированные списки данного атрибута. Запись индекса значения, указывающая на блок с инвертированным списком, начало которого находится в предыдущем блоке, вместо младшего BCH-списка этого блока содержит 0. Записи индекса значения, относящиеся к определенному атрибуту, объединяются в блоки. В блоке может находиться только целое число записей. Способ выделения новых блоков тот же, что и для инвертированных списков.

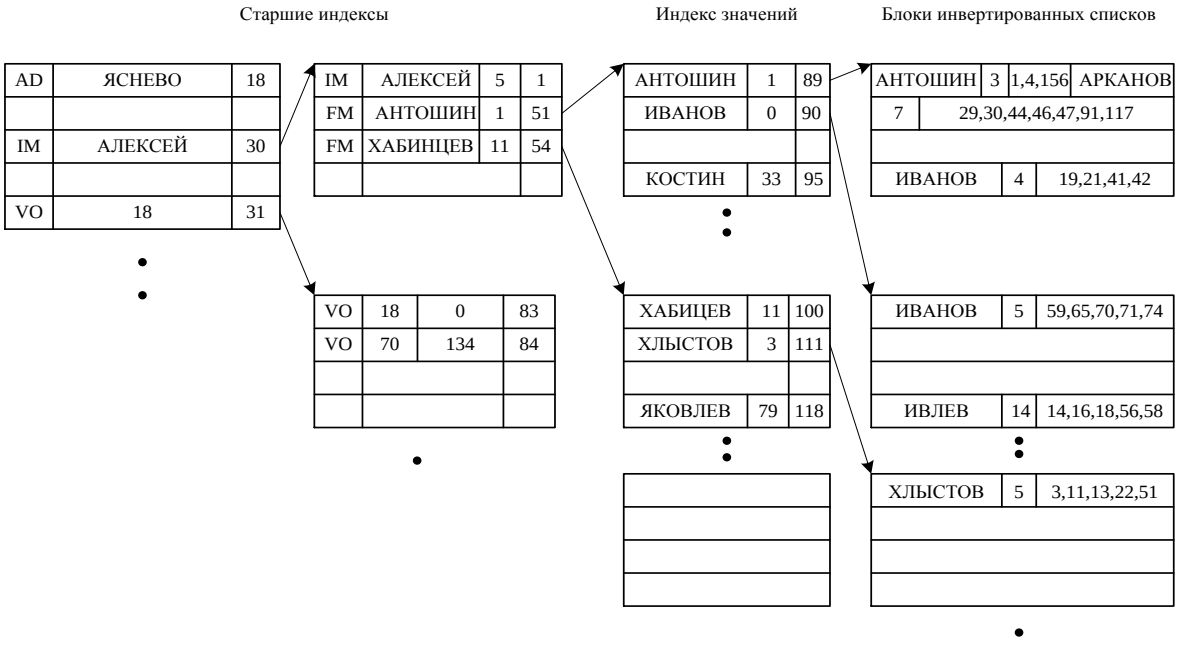


Рис 2.4. Инвертированные структуры

Нужный блок индекса значения, относящегося к данному атрибуту, отыскивается посредством старшего индекса. Блок старшего индекса состоит из записей. Каждая запись адресует один блок индекса значений. Запись включает код имени атрибута, поле признаков атрибута, длину значений атрибута, BCH и адрес блока индекса значения. На все поисковые атрибуты заведены записи в старшем индексе. Поисковые атрибуты, не имеющие значений в записях файла,

представлены в записях старшего индекса нулевыми значениями полей длины, ВСН и адреса блока индекса значений. В поле признаков атрибута указываются характеристики атрибута: формат хранения, принадлежность к повторяющейся группе, признак множественного атрибута.

Каждая запись содержит указатель на один блок индекса значений. Количество блоков старшего индекса зависит от количества блоков индекса значений и, в конечном счете - от объема БД. В случае если количество блоков старшего индекса больше одного, то для поиска требуемого блока старшего индекса создается блок старшего индекса следующего уровня иерархии такой же структуры и т. д. Адрес самого верхнего уровня индекса для данного файла указывается в таблице размещения файла.

Связь ВСН записи с номером блока в накопителе, в котором находится эта запись, осуществляется посредством преобразователя адреса, который представляет собой таблицу, состоящую из трехбайтовых элементов. Элемент, относительный номер которого равен значению ВСН, содержит номер блока накопителя, в котором находится запись с этим ВСН. Размер таблицы определяется максимальным значением ВСН в данном файле, который задается как параметр при создании этого файла.

Схематически связь ВСН с блоками накопителя посредством преобразователя адреса показана на рис. 2.5.

Структура рабочего и вспомогательных наборов данных. Рабочий набор данных занимает один экстент памяти и состоит из блоков фиксированной длины. Он предназначен для временного размещения промежуточной информации, необходимой в процессе работы с БД. В составе рабочего набора выделяется область оперативного журнала изменений, используемого для поддержания логической целостности БД, области промежуточных списков ВСН, а также области результирующих списков ВСН.

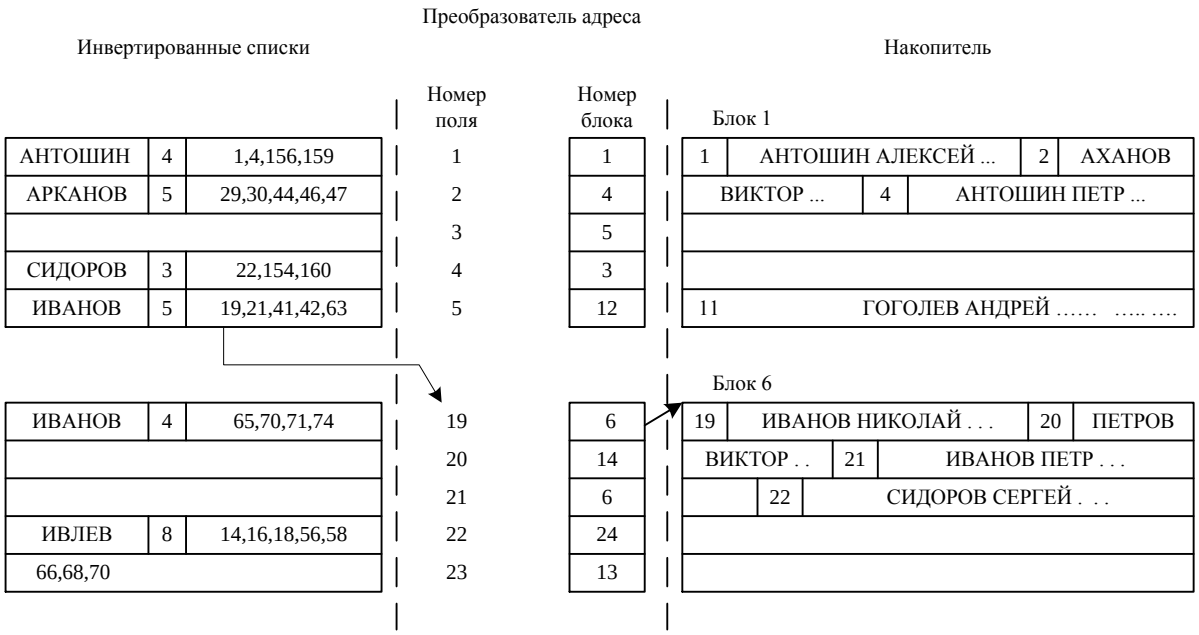


Рис. 2.5. Пример преобразования ВСН в адрес

Для работы ряда утилит предназначены вспомогательные наборы данных (набор для сортировки и временный набор данных). Эти наборы имеют прямую организацию с фиксированной длиной блока. Набор для сортировки используется при создании и модификации инвертированных списков. Временный набор данных применяется в качестве буферной памяти при загрузке БД для создания инвертированных списков. При функционировании системы эти наборы не используются.

2.1.3. Методы доступа к данным

Методы доступа к данным в существующих СУБД рассмотрены в [45, 150, 158-159, 161-165]. Операции доступа к данным СУБД ADABAS представлены операциями чтения и поиска записей файлов БД.

Операция чтения обеспечивает селекцию записи по ее позиции в файле или его части и выборку требуемых значений атрибутов, т. е. операция чтения эквивалентна последовательности операций селекции и выборки данных.

Операция поиска обеспечивает селекцию некоторой совокупности записей файла по значениям атрибутов этих записей или с учетом связей между файлами и, при необходимости, - выборку значений атрибутов из первой

отобранной записи. Частным случаем селекции по значениям атрибутов является рандомизированный доступ к записям файлов.

Чтение записей. В зависимости от вида упорядоченности множества записей операция чтения обеспечивает доступ к записям файла:

- по списку ВСН записей;
- в порядке возрастания ВСН записей файла;
- в логической последовательности по значениям заданного поискового атрибута;
- в физической последовательности расположения записей в БД.

Доступ к записи файла по списку ВСН осуществляется через системную таблицу, называемую преобразователем адреса. Преобразователь адреса организуется для каждого файла БД и отображает каждый ВСН в относительный номер блока набора данных, в котором размещается запись файла с этим ВСН.

При доступе к записям файла в порядке возрастания их ВСН возможно обращение к записи с минимальным ВСН, к записи с ВСН, ближайшим «большим» указанного, и обращение к записи со следующим (точнее, ближайшим — большим) ВСН независимо от физического размещения записей.

При доступе к записям в логической последовательности значений поискового атрибута в качестве упорядоченного множества, в котором производится селекция записей, выступает вся совокупность ассоциаций, построенных по значениям заданного поискового атрибута файла. Порядок в данном множестве определяется сортировкой полей индексных таблиц по значениям поисковых атрибутов и сортировкой каждого из инвертированных списков по значениям ВСН. В данном случае обеспечивается возможность обращения к первой записи (в смысле порядка, указанного выше), к конкретной записи и к следующей записи множества. Пример доступа к записям файла, логически упорядоченного по значениям

поискового атрибута, представлен на рис. 2.6. Селекция записей осуществляется с использованием инвертированных списков и позволяет просмотреть все записи ассоциаций записей, организованных по поисковому атрибуту ОК, начиная с заданной записи.

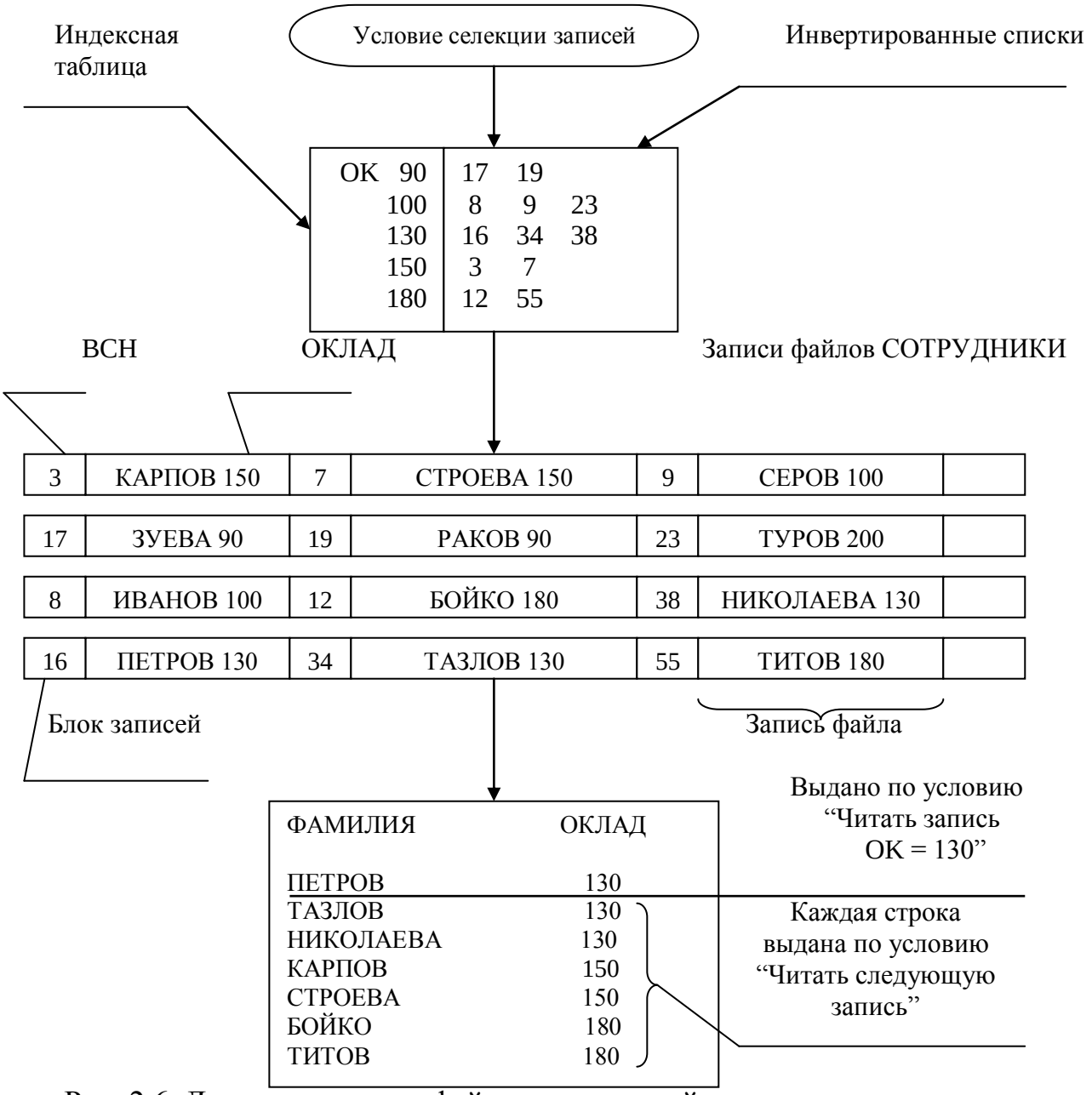


Рис. 2.6. Доступ к записям файла в логической последовательности по значениям поискового атрибута ОКЛАД (код имени ОК)

Доступ к записям файла в порядке физического размещения записей в БД основан на последовательном считывании блоков физической памяти файла в порядке их расположения в наборе данных. При этом возможно чтение первой

записи первого физического блока файла, чтение записи в указанной части памяти файла и чтение записи, физически следующей за указанной.

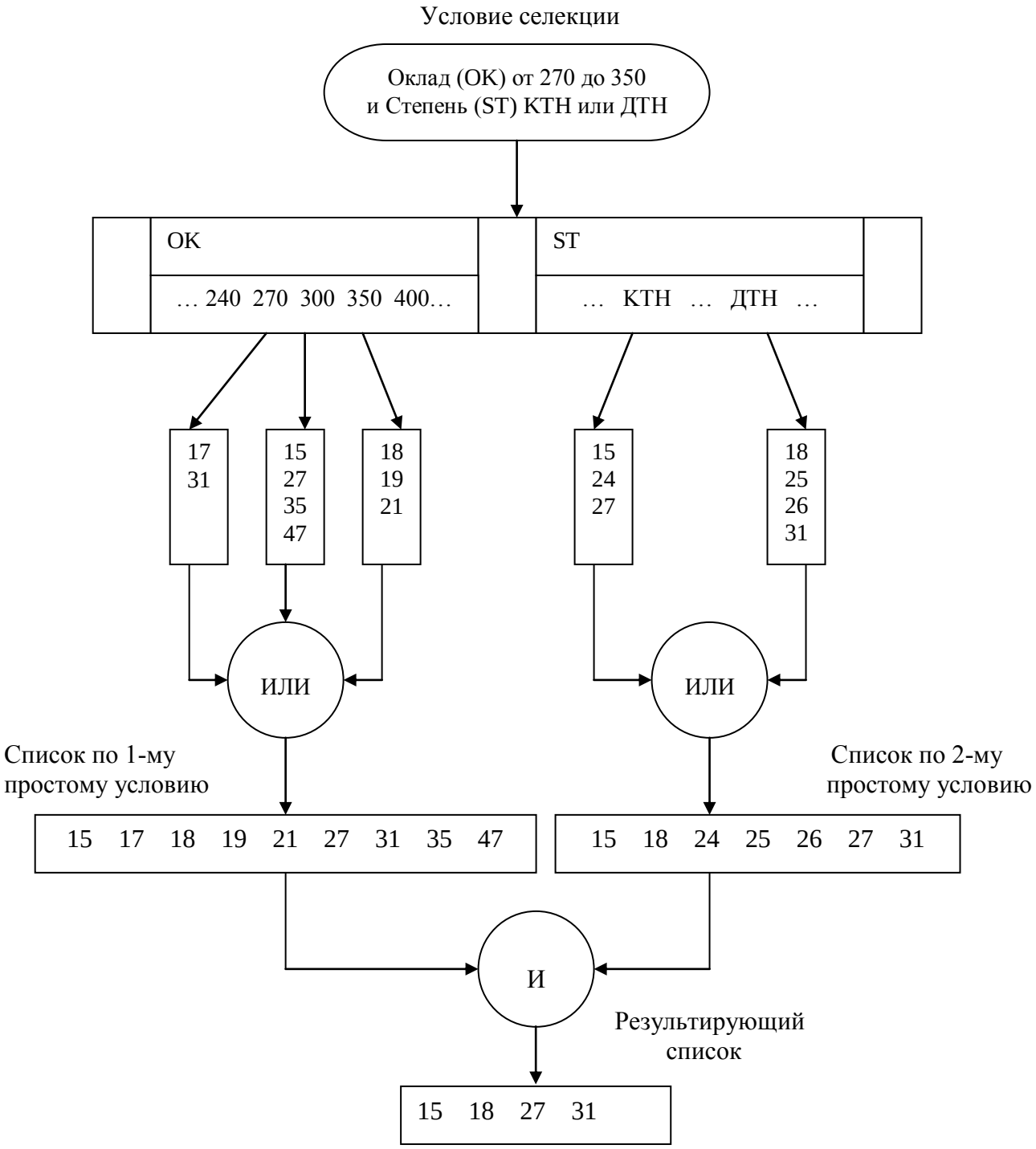


Рис. 2.7. Ассоциативный поиск записей файла

Поиск записей файла. Поиск записей в ADABAS, основанный на использовании ассоциаций, представленных в виде инвертированных списков и списков связи, определяется как ассоциативный поиск.

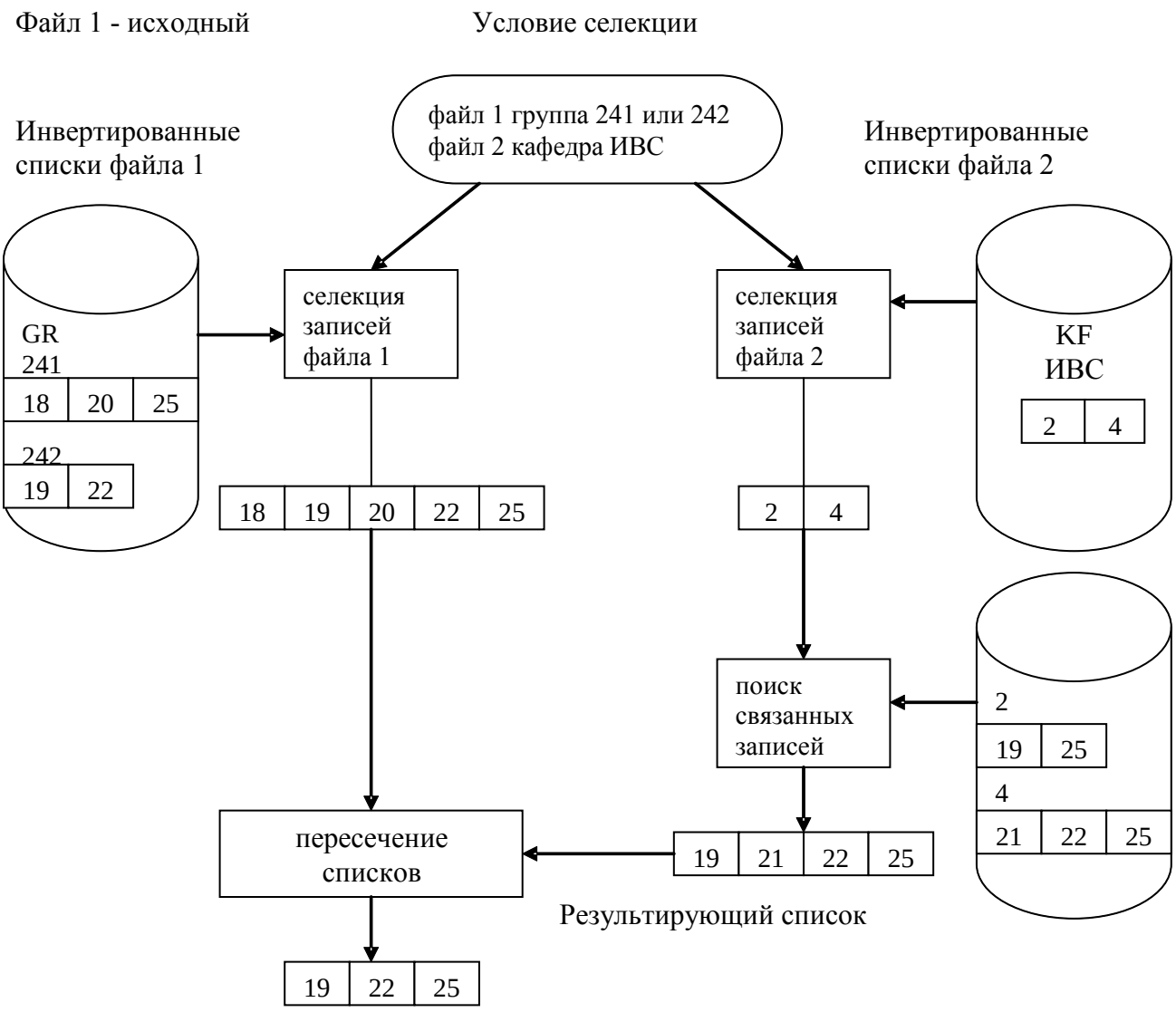
Ассоциативный поиск обеспечивает возможность селекции записей файла БД по значениям атрибутов, а также селекцию записей файла с учетом их связей с записями другого файла.

На рис. 2.7. представлена схема ассоциативного поиска с использованием инвертированных списков. Условие поиска состоит из двух простых условий, одно из них задано диапазоном значений атрибутов, а другое — перечнем значений. Результирующий список, полученный как пересечение двух списков, сформированных по простым условиям, содержит четыре ВСН записей, удовлетворяющих условию поиска.

Если полагать, что чтение каждого инвертированного списка с учетом многоуровневости индексных таблиц требует в среднем четырех операций ввода-вывода, то практически независимо от количества записей файла БД ассоциативный поиск в данном случае потребует 20 операций ввода-вывода, что дает ощутимый выигрыш во времени по сравнению с методами использования цепочек и прямого просмотра файла БД.

При использовании инвертированных списков некоторая зависимость времени поиска от количества записей файла проявляется в больших БД, содержащих 10^5—10^6 записей, поскольку возникает необходимость обработки больших списков ВСН по частям.

На рис. 2.8. представлена схема ассоциативного поиска в связанных файлах. Составное условие поиска в примере на рис. 2.8. соответствует требованию найти в файле 1 все записи о студентах учебных групп 241 и 242, участвующих в семинарах преподавателей кафедры информационно-вычислительных систем (ИВС). Файл 1 объявлен как исходный, поэтому результат поиска в виде списка из трех ВСН записей относится к файлу СТУДЕНТ.



2.8. Ассоциативный поиск записей в связанных файлах

Возможности ассоциативного поиска в БД могут быть расширены благодаря использованию операций над ассоциациями записей файлов. Данные операции обеспечивают возможность логической обработки и сортировки списков ВСН записей файла, которые могут быть получены в результате ассоциативного поиска или сформированы программой пользователя.

Операции логической обработки используют в качестве операндов два списка ВСН записей, принадлежащих одному файлу. Допускается три вида логических операций - пересечение списков, их объединение и пересечение с отрицанием.

Сортировка списка ВСН может быть выполнена в порядке возрастания значений ВСН в списке, а также в порядке возрастания или убывания значений от одного до трех поисковых атрибутов записей файла, которые представлены своими ВСН в сортируемом списке.

2.1.4. Ограничения целостности

Ограничения целостности СУБД ADABAS подробно рассмотрены в [45].

Ограничения целостности модели данных концептуального уровня ADABAS могут быть разделены на синтаксические и семантические.

Статические синтаксические ограничения, определяющие возможности ADABAS по отображению структурных свойств объектов предметной области и отношений между ними в схеме БД, выраженные через максимально допустимые значения параметров ЯОД, приведены ниже:

Число файлов в БД	255
Число записей файла	16,7*106
Число компонентов схемы файла	500
Число уровней в схеме файла	7
Число экземпляров повторяющейся группы	99
Число экземпляров множественного атрибута	191
Число поисковых атрибутов файла	200
Число элементов составного атрибута	5
Длина значения атрибута:	
– символьного, байт	253
– десятичного целого, разрядов	27
– упакованного, байт	14
– битового, байт	126
– двоичного целого, байт	4
– символьного или битового поискового, байт	126

− производного, байт	256
Число файлов, связанных с другим файлом	18

Динамическим синтаксическим ограничением целостности является запрет на использование в операции модификации более чем одной записи файла БД. Это ограничение минимизирует объем данных, передаваемых между прикладной программой и ADABAS в процессе ввода, корректировки или удаления записей файлов БД, что повышает динамичность работы ADABAS в режиме обслуживания многих пользователей.

Статические семантические ограничения целостности усиливают действие структурных ограничений, устанавливая дополнительные рамки на значения атрибутов в соответствии с семантикой моделируемых объектов предметной области и тем самым, ограничивая число возможных экземпляров объектов, представленных в БД. К данному виду ограничений можно отнести задание на ЯОД числа экземпляров множественного атрибута и периодических групп, а также задание длин значений атрибутов в пределах допустимых максимальных значений, определяемых структурными ограничениями. Возможность определения уникальных атрибутов в схеме файла также относится к числу статических семантических ограничений. Кроме того, статические семантические ограничения в ADABAS представлены возможностью контроля по словарю значений символьных атрибутов и контроля по диапазону значений для числовых атрибутов. Требование на указанные виды контроля определяется в словаре данных путем описания атрибутов, подлежащих контролю.

Динамические семантические ограничения, определяющие допустимые переходы БД из одного целостного состояния в другое в зависимости от свойств объектов предметной области, могут быть заданы путем использования специальных команд ЯМД. К таким командам относятся команды управления транзакциями, с помощью которых определяется допустимая

последовательность операций модификации данных, переводящая БД в новое целостное состояние.

2.2. Специфика среды проектирования приложений Natural

Чтобы быть способной реализовать программное и алгоритмическое наполнение программных модулей КИС лесопромышленного предприятия, а также осуществлять настройки программных модулей, среда разработки приложений должна обладать следующими возможностями:

1. Событийное программирование. Подразумевает выполнение определенной процедуры или подпрограммы на языке программирования среды разработки при совершении пользователем определенного действия, связанного с активными компонентами пользовательских приложений.

2. Функциональность расчетных операторов языка программирования среды разработки. Функциональность определяется наличием мощного языка программирования, включающего как операторы манипулирования массивами данных, так и ряд логических и математических функций.

3. Управление параметрами. Должна быть возможность исполнения различных программ на основе задаваемых параметров. Также должна быть предусмотрена процедура ввода и изменения значений параметров.

4. Иерархия приложений (приоритеты исполнения приложений и программ). Необходимо наличие средств назначения приоритетов перехода от приложения к приложению и порядка исполнения процедур различных приложений.

5. Возможность использования программ и запросов несколькими пользовательскими приложениями. В целях оптимизации расчетов должна обеспечиваться возможность вызова или исполнения каждой программы взаимодействия с БД, а также расчетной программы из любого из пользовательских приложений.

6. Стандартизация запросов и программ обработки. Для обеспечения доступа приложений к программам, необходимо каждый тип запроса к БД или расчета данных, сохраненного в виде стандартизированного файла.

Подробно описание среды редактора приложений Natural представлено в [45, 53, 55-59, 62, 34].

2.2.1. SPoD - единая система разработки прикладных приложений

Учитывая вышеизложенные требования к среде разработки приложений, для реализации КИС лесопромышленного предприятия предлагается наряду с СУБД ADABAS использовать продукты той же фирмы-производителя, объединенные под названием SPoD.

SpoD (Single Point of Development) – единая среда для разработки прикладных приложений. Благодаря данной технологии возможно разрабатывать приложения для различных платформ (Windows, Unix, Mainframe), для различных СУБД (Adabas, Tamino, Oracle, DB2 и пр.), а также с использованием различных технологий (DCOM, SOAP, Web Services, и пр.)

Перечень программных средств SPoD для разработки прикладных приложений, а также взаимосвязи между ними показаны на рис. 2.9.

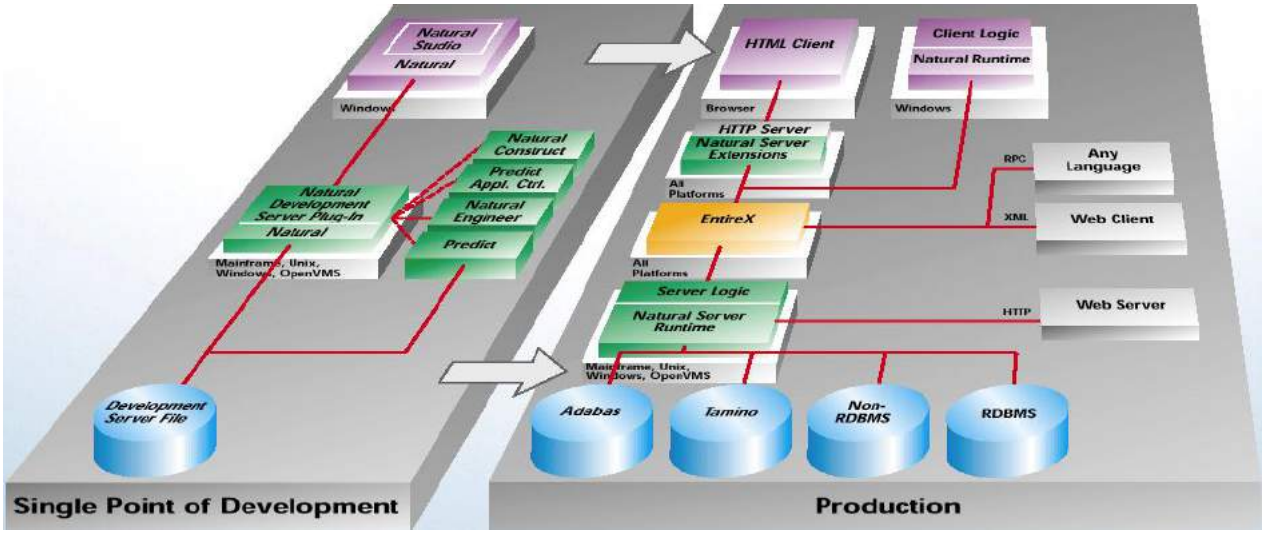


Рис. 2.9. Структура и взаимосвязи программных средств SPoD

SPoD удовлетворяет следующим требованиям и предлагает следующие преимущества:

– Клиент/сервер архитектуры, поддерживающий одну единственную удаленную среду разработки для всех платформ.

– Единый, знакомый образ всех объектов включаемых в прикладную разработку.

– Расширенная управляемая помощь/документация для дистанционных сред разработки.

– Увеличение производительности разработки через мощную графическую рабочую среду.

– Улучшение управления над заданиями разработки в Natural.

– Низкие издержки для разработки программного обеспечения, эксплуатации и администрирования.

Архитектура системы SpoD показана на рис. 2.10.

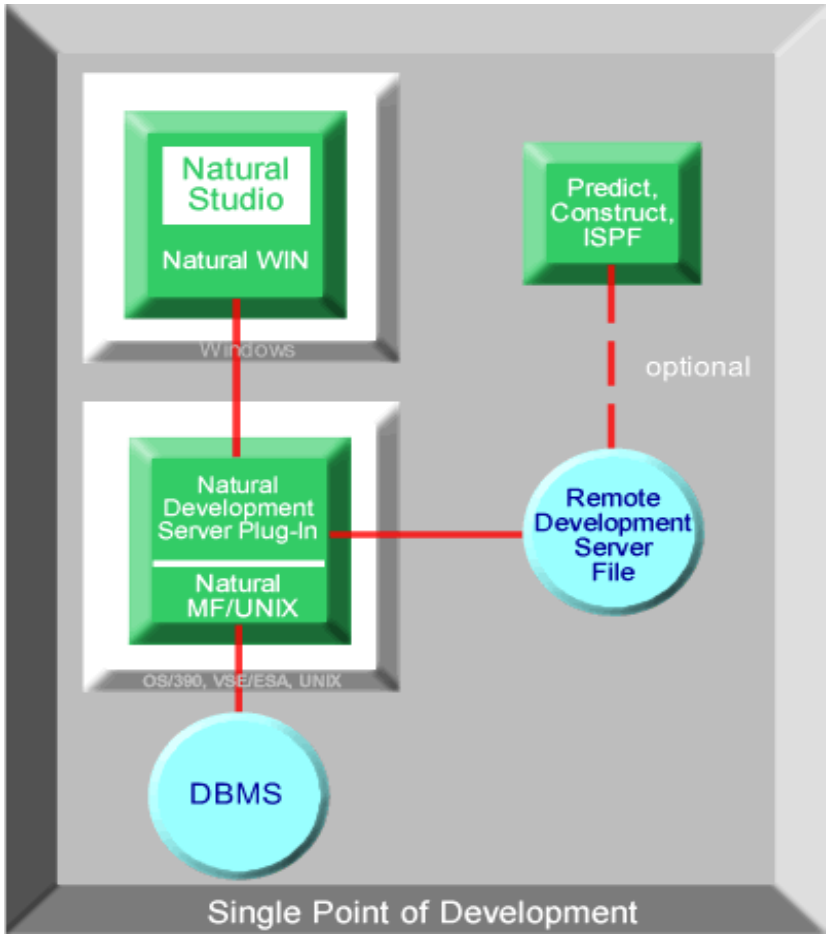


Рис. 2.10. Архитектура SPoD

Основными компонентами SPoD являются:

- Natural для Windows, как клиент удаленной разработки;
- Natural Development Server – сервер разработки;
- Development Server File – файл хранения данных;
- Predict and Natural Construct – идентичность файла хранения данных.

Natural для Windows, как клиент удаленной разработки. Компонент Natural Studio является рабочей станцией центральной разработки для всех платформ. Все Natural-связанные прикладные разработки, такие как: администрирование, удаленное редактирование, компиляция, отладка, настройку конфигурации и т.д., осуществляются из среды Natural Studio, независимо от платформы, где находится серверная среда разработки. Дистанционная связь клиента разработки с Natural на серверной платформе осуществляется через Natural Development Server посредством протокола TCP/IP.

Natural Development Server – сервер разработки. Средством удаленной разработки на серверной платформе является Natural с добавленным компонентом Natural Development Server (NDV).

Удаленный сервер разработки обеспечивает следующие функции:

- предоставление доступа к системным файлам;
- предоставление доступа к прикладным данным;
- согласование модификаций прикладных объектов посредством блокировки с использованием удаленного файла сервера разработки;
- выполнение дистанционных команд, задаваемых клиентом разработки Natural Studio.

Development Server File – файл хранения данных. Development Server File предназначен для хранения прикладных данных. Его структура подобна структуре системного файла Natural FDIC.

Predict and Natural Construct – идентичность файла хранения данных. В рамках SPoD такжу существует 2 дополнительных компонента: Predict - словарь данных и Natural Construct - генератор прикладных систем. В качестве файла сервера разработки может быть использован тот же самый файл, который используется в Predict и Natural Construct.

SpoD обладает следующими возможностями:

– Дистанционная обработка объекта. SPoD позволяет дистанционно манипулировать программными объектами, независимо от физического месторасположения.

– Дистанционное редактирование. SPoD позволяет извлекать исходные файлы из целевой среды, редактировать их на локальной рабочей станции, и затем сохранять их в целевой среде.

– Графический интерфейс пользователя. Все перечисленные редакторы работают в диалоговом режиме, что обеспечивает большие преимущества перед символьными редакторами.

– Дистанционная компиляция. Существует возможность запуска процесса компиляции с локальной рабочей станции в целевую среду.

– Дистанционная отладка ошибок. С помощью отладчика Natural Studio возможно устранить ошибки в приложениях, исполняемых как в целевой среде, так и на локальной рабочей станции.

– Блокировка объектов. Во время доступа к дистанционному серверу разработки, параллельные коррекции предотвращаются. При этом информация, вводимая при параллельной коррекции, держится в файле сервера разработки.

– Поддержка окна эмуляции терминалов. При эксплуатации универсальных приложений для испытания выхода на терминалы в Natural Studio предусмотрено окно эмуляции терминалов.

В рамках КИС лесопромышленного предприятия среда SPoD может выполнять следующие функции:

– Определение и генерация базы данных (ADABAS, и пр.).

– Разработка программ с использованием сервисов систем обработки транзакций (CICS, Complete).

– Генерация программ.

– Документирование приложений.

– Реинжениринг и поддержка работы приложений.

– Работа в среде Natural Studio с объектами информационной системы (очереди заданий, наборы данных и т.п.)

2.2.2. Средства взаимодействия приложений системы с СУБД ADABAS

Программы и алгоритмы взаимодействия с ядром системы КИС создаются посредством языка программирования среды разработки приложений.

Операторы языка среды разработки приложений взаимодействуют с утилитами СУБД, которые непосредственно выполняют поиск, чтение или выборку данных на основе задаваемых операторами критериев поиска. Широкий перечень функций, критериев и методов осуществления поиска, чтения, доступа и манипулирования данными операторами языка среды разработки за счет рационального их использования при построении запросов позволит сократить время на осуществление запросов и представление результатов в рамках пользовательского приложения.

Для обеспечения возможности использования преимуществ усовершенствованного метода комбинированных индексов, методов доступа и структур хранения данных вышеописанной СУБД при реализации КИС лесопромышленного предприятия требуется следующий перечень функций взаимодействия с БД:

– считывание записи файла БД в физической последовательности;

– считывание записи файла БД в логической последовательности;

– считывание значения поля БД;

– считывание записи с заданным последовательным внутренним системным номером записи в файле (ВСН);

– выборка набора записей из БД по критерию поиска, состоящему из полей, определенных как дескрипторы;

– добавление новой записи в файл БД;

– обновление записи в файле БД;

– удаление записи из файла БД;

– чтение данных транзакции, сохраненных предыдущим оператором;

– повторное чтение записи, обрабатываемой в настоящее время;

– задание и проверка пароля доступа пользователя к защищенному файлу;

– повторная попытка чтения записи, задержанной другим пользователем;

– выполнение обработки прерывания.

Кроме того, помимо вышеперечисленных функций взаимодействия с БД, необходим ряд логических и математических функций.

Используемый в SPoD, язык программирования содержит набор операторов, выполняющих все перечисленные функции взаимодействия, а также необходимый перечень операторов, обеспечивающих алгоритмическое и математическое наполнение приложений и расчетных программ.

Перечень операторов Natural и подробное описание синтаксиса и выполняемых ими функций представлено в [45, 53, 55-59, 62, 34].

2.2.3. Графический интерфейс и обработка отчетных форм

В целях представления данных в рамках пользовательских приложений КИС в наиболее понятном и удобном для конечного пользователя виде, приложения необходимо оформлять в максимально иллюстрированном виде. Также графический интерфейс служит для упрощения процедуры ввода данных (посредством использования полей со списком, полей выбора и пр. для типов заносимых данных, количество значений которых сравнительно невелико).

Ниже представлен перечень основных компонентов графического интерфейса:

– Область управления;

- Поле ввода;
- Групповой фрейм;
- Поле со списком;
- Кнопка;
- Ящик выбора;
- Надпись;
- Таблица;
- Компоненты ActiveX;
- OLE-объекты;
- Кнопка переключателя;
- Прочие.

При проведении основных расчетов, анализа и интерпретации данных модулями КИС лесопромышленного предприятия результаты оформляются и представляются в виде форм ведомостей и других документов. Таким образом, согласно реализуемой модели КИС (рис. 1.1.) компиляция форм должна осуществляться по запросу, осуществляемому в рамках пользовательского приложения, с последующим возвратом полученных отчетных форм в окно приложения для дальнейшей проверки и вывода на печать.

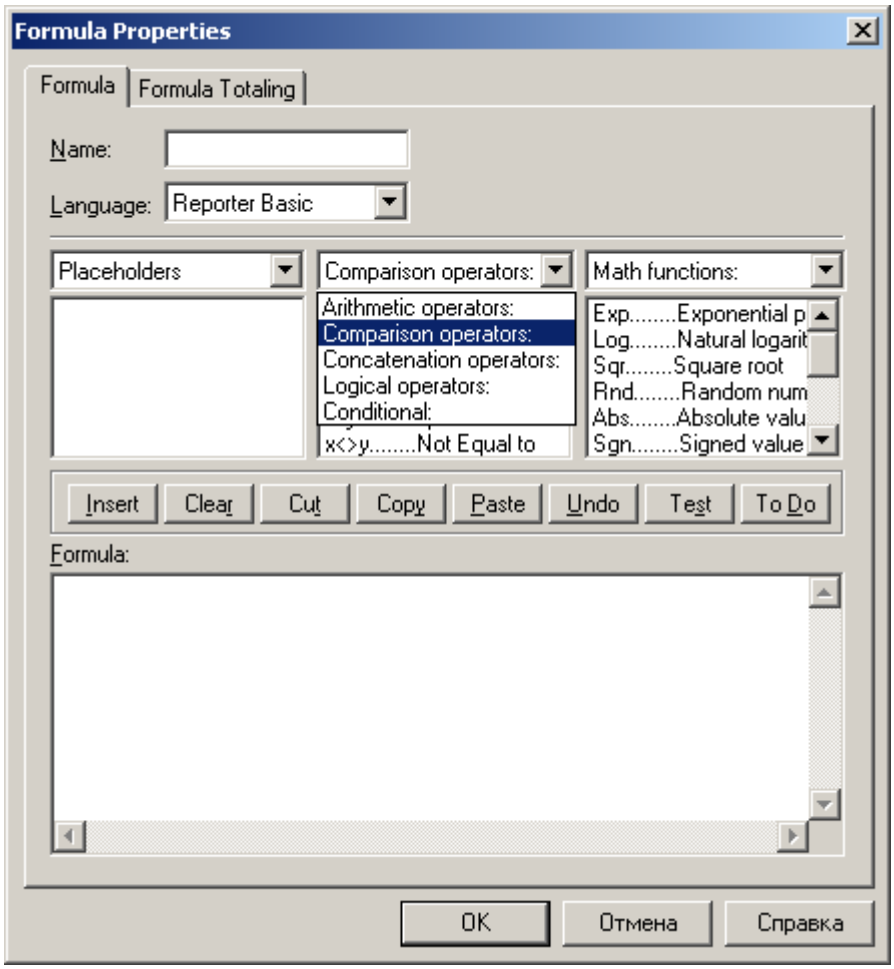


Рис. 2.11. Компилятор формул редактора Natural Reporter

В рамках среды SPoD данная концепция реализуется посредством встроенного редактора отчетных форм Natural Reporter. Natural Reporter также обладает мощным логистическим и математическим инструментарием для обработки анализируемых данных (рис. 2.11.)

2.3. Функционирование системы в условиях распределенной среды

Теория и свойства распределенных баз данных изложены в [41, 166-171].

Под распределенной базой данных (Distributed DataBase - DDB) обычно подразумевают базу данных, включающую фрагменты из нескольких баз данных, которые располагаются на различных узлах сети компьютеров. Для функционирования в условиях распределенной среды СУБД КИС приборостроительного предприятия должна соответствовать следующим 12 признакам:

1. Локальная автономия. Это качество означает, что управление данными на каждом из узлов распределенной системы выполняется локально. База данных, расположенная на одном из узлов, является неотъемлемым компонентом распределенной системы. Будучи фрагментом общего пространства данных, она, в то же время функционирует как полноценная локальная база данных. Управление базы данных осуществляется локально и независимо от других узлов системы.

2. Независимость от центрального узла. В идеальной системе все узлы равноправны и независимы, а расположенные на них базы являются равноправными поставщиками данных в единое информационное пространство. База данных на каждом из узлов самодостаточна, т.е. включает полный собственный словарь данных и полностью защищена от несанкционированного доступа.

3. Непрерывные операции. Это качество можно трактовать как возможность непрерывного доступа к данным в рамках распределенной системы вне зависимости от их расположения и вне зависимости от операций, выполняемых на локальных узлах.

4. Прозрачность расположения. Это свойство означает полную прозрачность расположения данных. Пользователь, обращающийся к распределенной БД, ничего не должен знать о реальном, физическом размещении данных в узлах информационной системы. Все операции над данными выполняются без учета их местонахождения. Транспортировка запросов к базам данных осуществляется встроенными системными средствами.

5. Прозрачная фрагментация. Это свойство трактуется как возможность распределенного размещения данных, логически представляющих собой единое целое. Существует фрагментация двух типов: горизонтальная (хранение строк одной логической таблицы в нескольких идентичных физических таблицах на различных узлах) и вертикальная (распределение столбцов

логической таблицы по нескольким узлам). Следующий пример иллюстрирует оба типа фрагментации. Имеется таблица employee (emp_id, emp_name, phone), определенная в базе данных на узле в городе A (city_A). Имеется точно такая же таблица, определенная в базе данных на узле в городе B (city_B). Обе таблицы хранят информацию о сотрудниках компании. Кроме того, в базе данных на узле в Далласе определена таблица emp_salary (emp_id, salary). Тогда запрос «получить информацию о сотрудниках компании» может быть сформулирован так:

READ ALL IN employee@city_A, employee@city_B ORDER BY emp_id

6. Прозрачность тиражирования. Тиражирование данных - это асинхронный процесс переноса изменений объектов исходной базы данных в базы, расположенные на других узлах распределенной системы. В данном контексте прозрачность тиражирования означает возможность переноса изменений между базами данных средствами, невидимыми пользователю распределенной системы.

7. Обработка распределенных запросов. Это свойство трактуется как возможность выполнения операций выборки над распределенной базой данных, сформулированных в рамках обычного запроса. То есть операцию выборки можно сформулировать с помощью тех же языковых средств, что и операцию над локальной базой данных. Например,

FIND customer.name, customer.address, order.number, order.date IN customer@london, order@paris WHERE customer.cust_number = order.cust_number

8. Обработка распределенных транзакций. Это качество можно трактовать как возможность выполнения операций обновления распределенной базы данных (STORE, UPDATE, DELETE), не разрушающее целостность и согласованность данных. Эта цель достигается применением двухфазового или двухфазного протокола фиксации транзакций (two-phase commit protocol), ставшего фактическим стандартом обработки распределенных транзакций. Его

применение гарантирует согласованное изменение данных на нескольких узлах в рамках распределенной (или, как ее еще называют, глобальной) транзакции.

9. Независимость от оборудования. Это свойство означает, что в качестве узлов распределенной системы могут выступать компьютеры любых моделей и производителей (от мэйнфреймов до персональных компьютеров).

10.Независимость от операционных систем. Это качество означает многообразие операционных систем, управляющих узлами распределенной системы.

11.Прозрачность сети. Доступ к любым базам данных может осуществляться по сети. Спектр поддерживаемых конкретной СУБД сетевых протоколов не должен быть ограничением системы с распределенными базами данных. Т.е. в распределенной системе обращение к БД может производиться по любым сетевым протоколам.

12.Независимость от СУБД. Это качество означает, что в распределенной системе могут сосуществовать СУБД различных производителей, и возможны операции поиска и обновления в базах данных различных моделей и форматов.

Все 12 перечисленных принципов, как показано в п.2.1. и п.2.2., в полной мере реализуются в СУБД ADABAS и взаимодействующей с ней среде разработки приложений SPoD.

2.4. Стоимость ПО среды проектирования и эксплуатации

Помимо технических и структурных характеристик СУБД и редакторов приложений, большое значение при принятии решения о приобретении некоторых программных продуктов для построения КИС лесопромышленного предприятия имеют оценка повышения уровня производительности после внедрения, экономического эффекта от внедрения, срока окупаемости инновации и т.д. Таким образом, при разработке и использовании автоматизированных систем управления встает проблема оценки стоимости ее использования.

Сравнительный анализ затрат на использование 3-х ведущих СУБД показан в [172].

На основании проведенных в [172] расчетов, был сделан вывод, что при оценке по критерию «технические возможности – стоимость использования» наилучшей средой разработки КИС лесопромышленного предприятия является СУБД ADABAS и редактор приложений Natural.

2.5. Выводы

1. Вышеописанные структура хранения данных и методы доступа к данным СУБД ADABAS позволяют использовать наиболее эффективный метод доступа к данным в условиях автоматизации деятельности лесопромышленного предприятия – усовершенствованный метод комбинированных индексов.

2. Вышеописанные структура хранения данных и методы доступа к данным СУБД ADABAS обеспечивают наиболее производительный способ получения статистической информации о хранящихся в БД данных – путем задания поисковых полей (дескрипторов).

3. Многоуровневая структура хранения данных (использование периодических групп и множественных полей) СУБД ADABAS позволяет создавать модель структуры лесопромышленного предприятия и модель структуры КИС лесопромышленного предприятия.

4. Мощный инструментарий среды разработки приложений SPoD позволяет конструировать сложные пользовательские приложения, программы взаимодействия с ядром КИС, расчетные программы, отчетные формы, а также, задавать взаимосвязи между ними.

5. Возможность работы СУБД ADABAS и среды разработки приложений SPoD в условиях распределенной среды позволяет создавать крупные корпоративные системы, подразделения которых могут находиться на больших расстояниях друг от друга.

6. Выбранная среда для реализации КИС лесопромышленного предприятия является наиболее подходящей при оценке по критерию «технические возможности – стоимость использования».

3. СИСТЕМНЫЕ СВЯЗИ И ФУНКЦИОНИРОВАНИЕ КОМПОНЕНТОВ КОРПОРАТИВНОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ УПРАВЛЕНИЯ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

3.1. Уровень организации ядра КИС лесопромышленного предприятия

При создании КИС лесопромышленного предприятия необходимо учитывать имеющиеся связи между структурными подразделениями, иерархию полномочий и их делегирование, а также значимость этих связей при осуществлении различных процессов деятельности.

В рамках теории управления рассматривается несколько возможных форм представления структуры организации. Все они, как правило, сводятся к графическим отображениям, когда при помощи символов, обозначающих различные подразделения, а также схематических связей, обозначающих отношения между подразделениями (количественных и качественных), строится адекватная модель структуры организации. При этом любая организация может быть представлена с разной степенью подробности (вплоть до отображения мельчайшей административной единицы, существующей в организации).

Графически отображенная модель структуры организации необходима при принятии стратегических и оперативных решений по реструктуризации организации с целью более эффективного управления, по созданию дополнительных подразделений, при реорганизации отношений (финансовых, и пр.) при желаемом увеличении прибыли и т.д.

В рамках КИС лесопромышленного предприятия, создание модели структуры также необходимо. Модель структуры в данном случае создается посредством соответствующей организации ядра КИС. При этом способы задания модели структуры могут быть различными.

В среде СУБД ADABAS все связи между переменными могут быть представлены в следующих видах:

1. Создание структуры при помощи задания свойств полей в БД, фиксированные связи между которыми могут отражаться в виде той или иной иерархической структуры, которая может задаваться различными способами. Вся последующая обработка данных, а также интерфейс предоставления конечных результатов (результаты анализа, учета и пр.) происходит при помощи средств Natural. При этом сама структура в понятной форме будет отражать действительную структуру организации. Структура БД, таким образом, будет являться самой моделью структуры.

2. Создание структуры путем задания параметров переменных в БД без отображения иерархии структуры. Сама структура при этом будет вырисовываться только при обработке данных средствами Natural и при последующем графическом отображении (также с помощью средств Natural).

В зависимости от уровня гибкости структуры организации при создании ядра КИС лесопромышленного предприятия в СУБД ADABAS предлагается использовать следующие типы структуры ядра (пп.3.1.1.-3.1.5.).

3.1.1. Основные аспекты организации ядра в СУБД ADABAS

Структура ядра модуля модификации задается при помощи указания уровней полей, объединения полей в группы и периодические группы.

Объединение полей в группы и периодические группы производится объявлением вспомогательного поля определенного уровня группой или периодической группой в рамках создаваемой таблицы определения данных (таблица FDT). При этом все последующее поля, определенные со следующим уровнем по отношению к уровню группы, считаются элементами группы. Каждой строкой определяется один элемент структуры ядра модуля модификации. Каждая строка состоит из следующих базисных характеристик:

1, A1, 20, A,

где

1 – номер уровня элемента;

A1 – наименование элемента (всегда двухбайтное);

20 – длина элемента (в байтах);

A – формат (в данном случае – алфавитно-цифровой; также существуют двоичный, шестнадцатеричный, упакованный и др. форматы).

Ниже показан пример объединения полей в группы:

1 , AA

2 , A1, 1, A

2 , A2, 4, A

В данном примере элемент AA имеет первый уровень и определен в качестве группы. Т.е. элементы 2-го уровня A1 и A2 являются элементами группы AA.

3.1.2. Реляционная структура ядра

Существует два подхода к проектированию реляционной базы данных.

– Первый подход заключается в том, что на этапе концептуального проектирования создается не концептуальная модель данных, а непосредственно реляционная схема БД, состоящая из определений реляционных таблиц, подвергающихся нормализации.

– Второй подход основан на механическом преобразовании функциональной модели, созданной ранее, в нормализованную реляционную модель. Этот подход чаще всего используется при проектировании больших, сложных схем баз данных, необходимых для корпоративных информационных систем.

На рис. 3.1. представлен пример реляционной модели (отношение степени 5) в виде таблицы, содержащей некоторые сведения о работниках гипотетического предприятия. Строки таблицы соответствуют кортежам. Каждая строка фактически представляет собой описание одного объекта реального мира (в данном случае работника), характеристики которого содержатся в столбцах. Можно провести аналогию между элементами реляционной модели данных и элементами модели "сущность-связь".

Реляционные отношения соответствуют наборам сущностей, а кортежи - сущностям. Поэтому, также как и в модели "сущность-связь" столбцы в таблице, представляющей реляционное отношение, называют атрибутами.

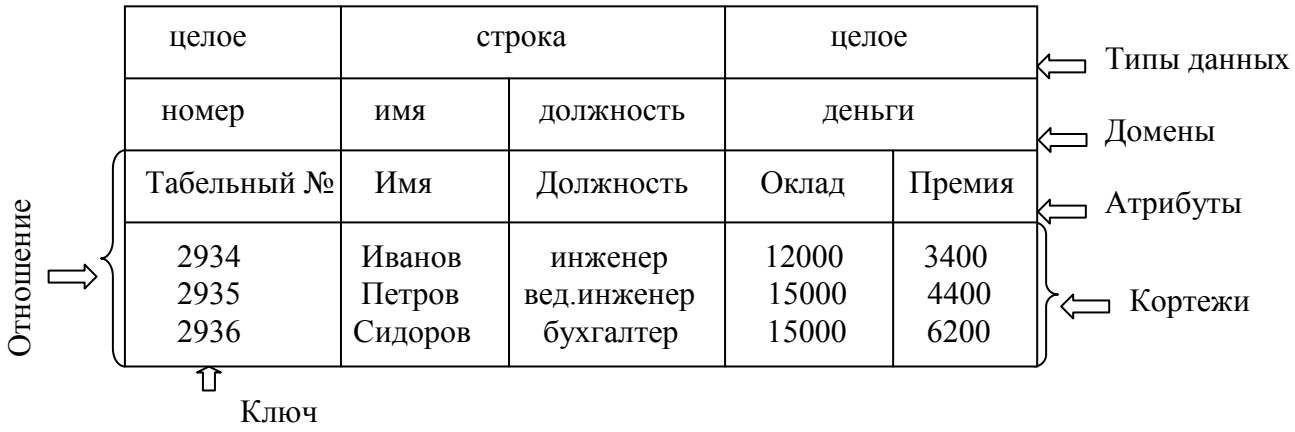


Рис.3.1. Основные компоненты реляционного отношения

В реляционной модели отсутствует понятие группового отношения. Для отражения ассоциаций между кортежами разных отношений используется дублирование их ключей. Пример реляционной модели базы данных, содержащей сведения о подразделениях предприятия и сотрудниках, показан на рис. 3.2.

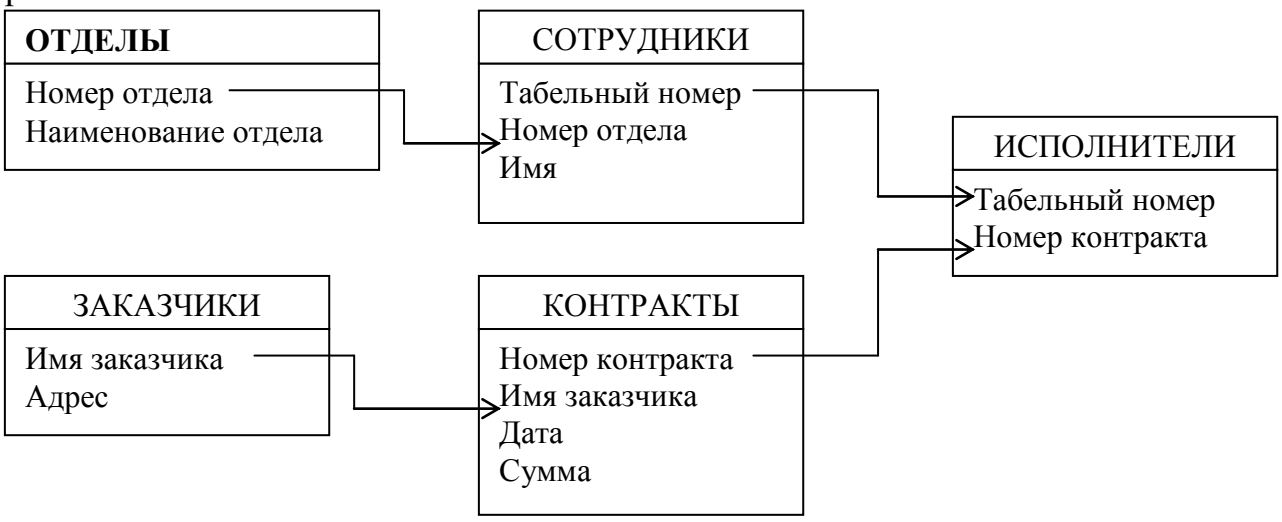


Рис.3.2. База данных о подразделениях и сотрудниках предприятия

В рамках СУБД ADABAS реляционная модель реализуется двумя способами:

1. Для каждого отношения (таблицы) – создание отдельного файла БД. Представленная выше БД о подразделениях и сотрудниках в СУБД ADABAS будет реализована в следующем виде:

File 1 (соответствует таблице ОТДЕЛЫ):

1 , A1, 2, A, DE - номер отдела

1 , A2, 40, A - наименование отдела

File 2 (соответствует таблице СОТРУДНИКИ):

1 , A1, 2, A, DE - номер отдела

1 , B1, 10, A, DE - табельный номер

1 , C1, 50, A - имя

File 3 (соответствует таблице ИСПОЛНИТЕЛИ):

1 , B1, 10, A, DE - табельный номер

1 , D1, 10, A, DE - номер контракта

File 4 (соответствует таблице ЗАКАЗЧИКИ):

1 , F1, 50, A, DE - имя заказчика

1 , G1, 50, A - адрес заказчика

File 5 (соответствует таблице КОНТРАКТЫ):

1 , F1, 50, A, DE - имя заказчика

1 , D1, 10, A, DE - номер контракта

1 , H1, 8, A - дата

1 , I1, 10, U - сумма

2. Создание одного файла БД для основного отношения (для хранения полной информации о каждом контракте) и создание трех дополнительных файлов (для хранения справочной информации об отделах, сотрудниках, заказчиках). Представленная выше БД о подразделениях и сотрудниках в СУБД ADABAS будет иметь следующую структуру:

File 1 – основное отношение:

1 , A1, 2, A, DE - номер отдела

1 , B1, 10, A, DE - табельный номер

1 , F1, 50, A, DE - имя заказчика

1 , D1, 10, A, DE - номер контракта

1 , H1, 8, A - дата

1 , I1, 10, U - сумма

File 2 - для хранения справочной информации об отделах:

1 , A1, 2, A, DE - номер отдела

1 , A2, 40, A - наименование отдела

File 3 - для хранения справочной информации о сотрудниках:

1 , B1, 10, A, DE - табельный номер

1 , C1, 50, A - имя

File 4 - для хранения справочной информации о заказчиках:

1 , F1, 50, A, DE - имя заказчика

1 , G1, 50, A - адрес заказчика

3.1.3. Иерархическая структура ядра

Компоненты базы данных, основанной на иерархической модели, могут быть представлены в виде рис. 3.3.

Основная единица обработки - запись. К основным понятиям иерархической структуры относятся: уровень, элемент (узел), связь.

Узел - это совокупность атрибутов данных, описывающих некоторый объект. На схеме иерархического дерева узлы представляются вершинами графа. Каждый узел на более низком уровне связан только с одним узлом, находящимся на более высоком уровне. Иерархическое дерево имеет только одну вершину (корень дерева), не подчиненную никакой другой вершине и находящуюся на самом верхнем (первом) уровне. Зависимые (подчиненные) узлы находятся на втором, третьем и т.д. уровнях. Количество деревьев в базе данных определяется числом корневых записей.

К каждой записи базы данных существует только один (иерархический) путь от корневой записи.

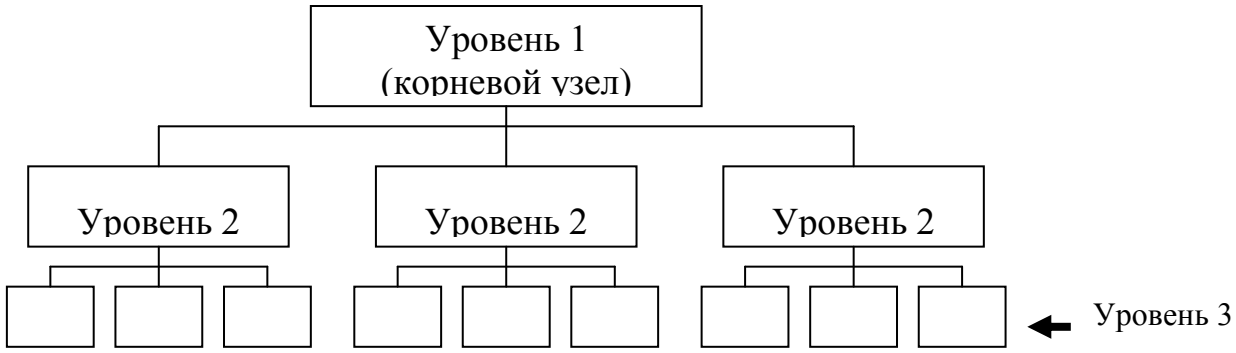


Рис 3.3. Древовидная структура компонентов баз данных, основанных на иерархической модели хранения данных

В рамках СУБД ADABAS иерархическая модель БД о подразделениях и сотрудниках предприятия (рис. 3.2.) реализуется следующим способом:

File 1 (реализующий связь ОТДЕЛЫ-СОТРУДНИКИ-ИСПОЛНИТЕЛИ):

1 , A1, 2, A, DE - номер отдела

1 , A2, 40, A - наименование отдела

1 , AA - группа СОТРУДНИКИ (2-й уровень)

2 , B1, 10, A, DE - табельный номер

2 , B2, 50, A - имя

2 , BB - группа ИСПОЛНИТЕЛИ (3-й уровень)

3 , C1, 10, A, DE - номер контракта

File 2 (реализующий связь ЗАКАЗЧИКИ-КОНТРАКТЫ-ИСПОЛНИТЕЛИ):

1 , A1, 50, A, DE - имя заказчика

1 , A2, 50, A - адрес заказчика

1 , AA - группа КОНТРАКТЫ (2-й уровень)

2 , B1, 10, A, DE - номер контракта

2 , B2, 8, A - дата

2 , B3, 10, U - сумма

2 , BB - группа ИСПОЛНИТЕЛИ (3-й уровень)

3 , C1, 10, A, DE - табельный номер

3.1.4. Многоуровневая структура ядра

Основными отличиями многоуровневой структуры ядра от иерархической являются:

– возможность задания нескольких вершин (корней дерева);

– в каждой из корневых записей может присутствовать несколько реализаций, что позволяет использовать связь M:N.

Остальные связи и правила создания иерархических структур в СУБД ADABAS остаются теми же и при создании многоуровневой структуры. Основные компоненты БД, основанные на многоуровневой модели структуры ядра, представлены на рис 3.4.

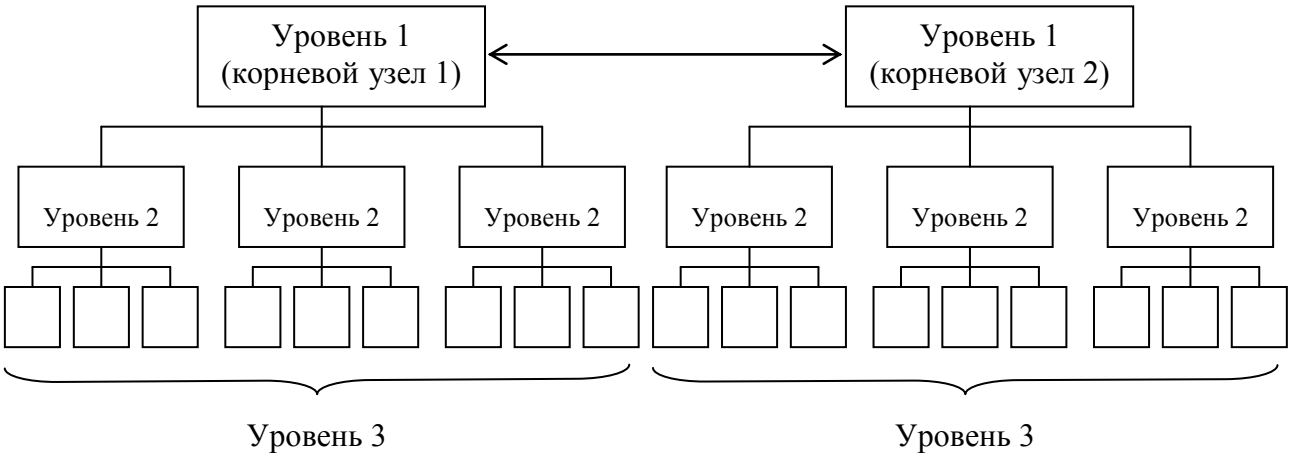


Рис. 3.4. Структура компонентов баз данных, основанных на многоуровневой модели структуры ядра

В рамках СУБД ADABAS вышеописанная модель структуры ядра создается при помощи объединения полей в периодические группы. Периодическая группа может состоять из одного или более полей и в пределах данной записи могут встречаться от 0 до 191 реализаций, но, по крайней мере, одна реализация (даже, если она содержит все нулевые значения) должна присутствовать в каждой входной записи.

Периодическая группа должна быть описана на уровне 1. Все поля, которые должны содержаться в периодической группе, должны следовать

тотчас после и должны быть описаны на уровне 2 или выше (с приращением 1 до максимального 7 уровня). Следующее описание уровня 1 указывает на конец текущей периодической группы.

В СУБД ADABAS многоуровневая структура БД о подразделениях и сотрудниках предприятия (представленная на рис. 3.2.) реализуется следующим способом:

File 1 (реализующий связи ОТДЕЛЫ-СОТРУДНИКИ-ИСПОЛНИТЕЛИ и ЗАКАЗЧИКИ-КОНТРАКТЫ-ИСПОЛНИТЕЛИ):

1 , AA, PE	- периодическая группа ОТДЕЛЫ-СОТРУДНИКИ-ИСПОЛНИТЕЛИ
2 , AB	- группа ОТДЕЛЫ (2-й уровень)
3 , A1, 2, A, DE	- номер отдела
3 , A2, 40, A	- наименование отдела
3 , AC	- группа СОТРУДНИКИ (3-й уровень)
4 , B1, 10, A, DE	- табельный номер
4 , B2, 50, A	- имя
4 , AD	- группа ИСПОЛНИТЕЛИ (4-й уровень)
5 , C1, 10, A, DE	- номер контракта
1 , BA, PE	- периодическая группа ЗАКАЗЧИКИ-КОНТРАКТЫ-ИСПОЛНИТЕЛИ
2 , BB	- группа ЗАКАЗЧИКИ (2-й уровень)
3 , D1, 50, A, DE	- имя заказчика
3 , D2, 50, A	- адрес заказчика
3 , BC	- группа КОНТРАКТЫ (3-й уровень)
4 , F1, 10, A, DE	- номер контракта
4 , F2, 8, A	- дата
4 , F3, 10, U	- сумма
4 , BD	- группа ИСПОЛНИТЕЛИ (4-й уровень)
5 , G1, 10, A, DE	- табельный номер

3.1.5. Мультипольная структура ядра

Характерной чертой мультипольной структуры ядра БД является возможность хранения нескольких значений в рамках одного, специальным способом определенного, поля. В СУБД ADABAS такое поле носит название «мультиполе» (или множественное поле). Фактическое количество значений, присутствующих в каждой записи мультиполя, может меняться от 0 до 191, но, по крайней мере, одно значение (даже пустое) должно присутствовать в каждой входной записи.

Значения хранятся согласно другим опциям, указанным для поля. Первое значение - это предшествующий полю счетчик, который указывает количество значений, в настоящее время присутствующих в поле. Количество хранимых значений, равное количеству значений, представленных во входной записи, плюс любые значения, добавленные во время обновления поля, меньше любых подавленных значений (в случае, если поле определено с опцией подавления пустых значений).

Если количество значений, содержащихся в каждой входной записи постоянно, это количество может быть определено при определении мультиполя в формате MU(n), где "n" равняется количеству значений, представленных в каждой входной записи. Если количество значений не постоянно для всех входных записей, то однобайтовый двоичный счетчик поля должен предшествовать первому значению каждой входной записи, указывая количество существующих в ней значений.

Преимущества использования мультипольных структур проявляются в тех случаях, когда необходимо для каждой записи периодически вносить дополнительные данные. Например, в случае с представленной на рис. 3.2. БД использование мультиполей было бы целесообразно при ведении учета ежемесячно начисляемой заработной платы для каждого из сотрудников. В рамках реляционной БД эта связь реализовывалась бы в следующем виде (табл. 3.1):

Структура реляционной БД начисления заработной платы сотрудникам

Таблица 3.1.

Табельный номер сотрудника	Номер отдела	Дата начисления заработной платы	Сумма
2457578765	06	02.01.08	18000
2457578765	06	02.02.08	16700
2457578765	06	02.03.08	19200

По каждому начислению заработной платы в таблице БД создается новая запись (табл. 3.1.). При этом связь с другими таблицами БД (например, ИСПОЛНИТЕЛИ и ОТДЕЛЫ) осуществляется посредством ключей «Табельный номер» и «Номер отдела».

При использовании мультиполей, если представить мультиполную структуру в виде двухмерной таблицы, ее вид будет следующим (табл. 3.2.):

Структура мультипольной структуры БД начисления заработной платы сотрудникам

Таблица 3.2.

Табельный номер сотрудника	Номер отдела	Дата начисления заработной платы	Сумма
2457578765	06	02.01.08	18000
		02.02.08	16700
		02.03.08	19200

В данном случае при начислении заработной платы не создается новая запись целиком, а только добавляются соответствующие значения в мультиполя («Дата начисления заработной платы» и «Сумма» - табл. 3.2.). При этом данная структура может быть как самостоятельным файлом БД, так и частью другого файла любой из вышеописанных структур ядра.

В СУБД ADABAS мультипольная структура БД начисления заработной платы (табл. 3.2.) реализуется следующим способом:

File 1 (основное отношение):

1 , A1, 10, A, DE - табельный номер сотрудника

1 , A2, 2, A, DE - номер отдела

1 , A3, 8, A, MU - дата начисления заработной платы

1 , A4, 7, U, MU - сумма

3.1.6. Смешанная структура ядра

Организация смешанной структуры ядра КИС представляет синтез свойств структур ядра, описанных в пп.3.1.2.-3.1.5.

Мультимодельность ADABAS в совокупности с рядом дополнительных возможностей, позволяет строить как сугубо традиционные иерархические, сетевые и реляционные SQL базы данных, так и сложные текстовые информационно-поисковые и интегрированные системы и системы обработки изображений, постреляционные структуры для моделирования человеческой деятельности, экспертного анализа сложных производственных процессов и т.д. При этом можно сочетать преимущества различных подходов. Структуры ядра всех типов могут беспрепятственно использоваться как при создании нескольких файлов различных структур, так и в рамках одного файла.

В СУБД ADABAS можно спроектировать БД в третьей нормальной форме и при этом, в целях повышения производительности часть связей преобразовать в иерархию. Таким образом, проектировщику предоставляется возможность выйти за рамки ограничений определенного подхода и, объединив преимущества всех подходов, достичь большей производительности и гибкости создаваемой системы на конкретных запросах. Все многообразие видов информационных систем и технологий становится возможным благодаря тому, что ADABAS обеспечивает поддержку следующих моделей (и типов) данных:

− Непервая нормальная форма (NF2 - Non-First Normal Form). Традиционная реляционная модель данных. Эта модель соответствует ANSI/ISO стандарту SQL92 и реализована в виде либо надстройки над ADABAS, либо как неотъемлемая часть ADABAS D.

− Модель данных сущность-связь (E/R модель). В ADABAS предусмотрено расширение до E/R модели (Entity-Relationship модель) для

управления сложными структурами данных с высокой степенью связности, а также рекурсивные структуры данных (когда элемент структуры данных содержит один или несколько указателей на элементы такого же типа). Объединяя эту модель с другими моделями ADABAS можно строить чрезвычайно мощные интегрированные базы данных и, соответственно, прикладные системы. Предпочтительные области для применения E/R моделей — системы представления знаний, моделирование поведения сложных технических и биологических систем, расчеты потребностей, планирование материальных ресурсов различного вида и назначения (Bills of Materials). Например, традиционные для последней предметной области проблемы информационного взрыва и управления циклами легко разрешимы с помощью возможностей E/R расширения ADABAS.

– Обработка и управление произвольными текстами. Этот тип данных (и соответствующие средства манипулирования ими) обеспечивает доступ к документам, обрабатываемым ADABAS, как к произвольным текстам.

– Изображения и аудиоинформация. Поддержка изображений и аудиоинформации в ADABAS основана на функции гипердескрипции (hyperdescriptor). Она позволяет использовать множественные значения индексов, выработанных внешним матобеспечением, или определенных пользователем, для концептуально однородных объектов (тезаурус). Эти значения, собственно, и обрабатываются далее ADABAS, делая, таким образом, возможным обращение к объектам и работу с ними при помощи других процедур, предоставляемых СУБД.

ADABAS имеет мощные и удобные в работе средства администрирования баз данных, реализованные как в интерактивном, так и в пакетном режимах, на всех серверных платформах.

На основе ADABAS в мире построено множество больших прикладных систем, как OLTP, так и OLAP. В то же время, благодаря возможности

испоьзования смешанных структур ядра БД, ADABAS может служить основой не только "традиционных" БД, но и хранилищ данных (Data Warehouse).

3.1.7. Рекомендации по выбору структуры ядра для КИС лесопромышленного предприятия

В таблице 3.3. в качестве примера показана часть БД КИС, отражающая структуру компонентов посменного учета расхода пиловочного сырья производственными подразделениями в рамках программного модуля управления производством КИС.

Структура ядра БД КИС для элементов посменного учета пиловочного сырья модуля управления лесопильным производством

Таблица 3.3.

№ п/п	Определение элементов (таблица FDT)	Название элемента	Соответствующий элемент структуры КИС лесопромышленного предприятия
1	1 , AM, 1, A	поле	Индикатор включения элементов программного модуля управления производством
2	1 , PM	группа	Программный модуль управления производством:
3	2 , AA, 1, A	поле	Индикатор включения элементов блока управления лесопильным производством
4	2 , AZ	группа	Блок управления лесопильным производством
5	3 , AP, 1, A	поле	Индикатор включения элементов посменного учета расхода пиловочного сырья
6	3 , A1	группа	Посменный учет расхода пиловочного сырья производственными подразделениями лесопильного производства.
7	4 , BP, 1, A	поле	Индикатор включения программ взаимодействия с ядром КИС
8	4 , B1, PE	периодическая группа	Программы взаимодействия с ядром КИС:

продолжение табл. 3.3.

9	5 , Z1, 1, A	поле	Программа интерпретации получаемых от приложений данных о посменном расходе сырья и внесения в БД
10	5 , Z2, 1, A	поле	Программа внесения плановых величин расхода пиловочного сырья
11	5 , Z3, 1, A	поле	Запрос количества израсходованного вида сырья за смену на определенную дату
12	5 , Z4, 1, A	поле	Запрос количества израсходованного вида сырья (порода, сорт, диаметр, длина) на определенную дату на определенной производственной стадии
13	5 , Z5, 1, A	поле	Запрос общего количества израсходованного сырья заданного сорта и древесной породы
14	5 , Z6, 1, A	поле	Запрос количества израсходованного сырья заданного сорта
15	5 , Z7, 1, A	поле	Запрос общего количества израсходованного сырья заданной длины и диаметра
16	5 , Z8, 1, A	поле	Запрос общего количества израсходованного сырья заданного диаметра
17	5 , Z9, 1, A	поле	Запрос продолжительности и причины простоя оборудования на заданной производственной стадии, заданным производственным подразделением на определенную дату
18	5 , ZA, 1, A	поле	Запрос продолжительности и причин простоя оборудования на всех производственных стадиях, всеми подразделениями на определенную дату и за промежуток времени
19	5 , ZB, 1, A	поле	Запрос количества сырья всех видов, израсходованного производственным подразделением за месяц
20	5 , ZC, 1, A	поле	Запрос количества сырья всех видов, израсходованного на производственных стадиях за месяц
21	5 , ZD, 1, A	поле	Запрос количества использованного сырья по критериям, выбираемым и задаваемым пользователем

продолжение табл. 3.3.

22	4 , BQ, 1, A	поле	Индикатор включения программ основных расчетов
23	4 , B2, PE	периодическая группа	Программы основных расчетов:
24	5 , Y1, 1, A	поле	Расчет стоимости пиловочного сырья, израсходованного каждым подразделением и всеми подразделениями за единицу времени
25	5 , Y2, 1, A	поле	Расчет процента выполнения плана расхода сырья, израсходованного каждым подразделением и всеми подразделениями
26	5 , Y3, 1, A	поле	Расчет стоимости сырья всех видов, израсходованного производственным подразделением за месяц
27	5 , Y4, 1, A	поле	Расчет стоимости сырья всех видов, израсходованного на производственных стадиях за месяц
28	5 , Y5, 1, A	поле	Расчет стоимости израсходованного за единицу времени, вид сырья, сорт и размеры которого выбирается пользователем
29	5 , Y6, 1, A	поле	
30	4 , BR, 1, A	поле	Индикатор включения пользовательских приложений
31	4 , B3, PE	периодическая группа	Пользовательские приложения:
32	5 , X1, 1, A	поле	Приложение внесения данных.
33	5 , X2, 1, A	поле	Приложение внесения плановых величин.
34	5 , X3, 1, A	поле	Приложение управления программным модулем КИС (управление запросами, расчетными программами, отчетными формами).
35	5 , X4, 1, A	поле	Приложение выбора критериев поиска и задания значений критериев
36	4 , BS, 1, A	поле	Индикатор включения отчетных форм
37	4 , B4, PE	периодическая группа	Отчетные формы:
38	5 , W1, 1, A	поле	Ведомость выполнения плана раскроя сырья

окончание табл. 3.3.

39	5 , W2, 1, A	поле	Ведомость объемов распиленного пиловочника
40	5 , W3, 1, A	поле	Ведомость простоя оборудования
41	5 , W4, 1, A	поле	Ведомость стоимости пиловочного сырья
42	5 , W5, 1, A	поле	Отчетные формы количества израсходованного сырья на основе критериев, задаваемых пользователем

При построении структуры ядра модуля модификации иерархия элементов КИС в таблице определения данных (таблице FDT) задается посредством следующих уровней:

1 – программные модули общего назначения (модули управления запасами, управления производством, планово-аналитической деятельности и т.д.)

2 – специализированные программные блоки – составляющие программных модулей общего назначения (для модуля управления производством лесопромышленного предприятия - блоки управления лесопильным производством, фанерным производством, производством древесных плит, мебельным производством, вспомогательными производствами).

3 – группы учета специализированных блоков (для блока управления лесопильным производством - посменный учет расхода пиловочного сырья производственными подразделениями лесопильного производства, посменный учет досок-полуфабрикатов произведенных подразделениями лесопильного производства, посменный учет расхода бревен для окорки окорочными подразделениями лесопильного производства и т.д.).

4 – сгруппированные компоненты КИС групп учета (это всегда программы взаимодействия с ядром системы, пользовательские приложения, программы основных расчетов, отчетные формы).

5 – непосредственно программы, алгоритмы, приложения, формы соответствующих компонентов КИС (запрос количества израсходованного вида сырья за смену на определенную дату, ведомость выполнения плана раскроя сырья и т.д.)

Также при помощи модуля модификации КИС лесопромышленного предприятия в файлах значений идентификаторов элементов записей (табл. 1.1.) задаются значения следующих переменных:

- Количество и наименования структурных подразделений предприятия;
- Количество хозрасчетных бригад, участков;
- Количество смен;
- Количество калькулируемых объектов (изделий);
- Количество и наименование деталей;
- Количество и наименование причин отклонения от нормы;
- Технические характеристики объекта:
    - Количество и наименования используемых древесных пород;
    - Назначение;
    - Количество и наименования способов распиловки;
    - Количество и наименования сортов;
- Типы клеильного пресса;
- Марки фанеры;
- Количество сушильных камер;
- Количество пакетов;
- Наименование причин простоя оборудования;
- Количество категорий качества сушки пиломатериалов;
- Количество режимов камерной сушки.

Для переменных, требующих определения дополнительных значений (наименования причин простоя оборудования и пр.) создается отдельный файл БД, и в пользовательском приложении предусматриваются соответствующие

элементы ввода дополнительных значений, которые активизируются пользователем.

Например, при редактировании переменных количества и наименований структурных подразделений лесопромышленного предприятия, требуется ввод количества подразделений, в зависимости от которого активизируется соответствующее количество полей ввода для занесения наименований каждого из подразделений. Структура файла значений идентификаторов элементов записей при редактировании переменных количества и наименований структурных подразделений показана ниже:

1 , AA, 2, A

1 , AB, PE

2 , A1, 100, A, MU

2 , A2, 2, A, MU, где

AA – количество структурных подразделений (поле);

AB – группа наименований структурных подразделений (периодическая группа);

A1 – наименования структурных подразделений (множественное поле);

A2 – порядковый номер структурного подразделение (множественное поле).

Многоуровневая структура, рассмотренная в примере, позволяет для каждой переменной (количество структурных подразделений – AA) хранить несколько значений (номер структурного подразделения и наименование).

Для определения значений переменных «технические характеристики объекта» часть структуры ядра модуля будет выглядеть следующим образом:

1 , BA, 2, A

1 , BB, PE

2 , B1, 2, A, MU

2 , B2, 100, A, MU

2 , B3, 150, A, MU

1 , B4, 2, A

1 , B5, PE

2 , C1, 2, A, MU

2 , C2, 100, A, MU

1 , B6, 2, A

1 , B7, PE

2 , D1, 2, A, MU

2 , D2, 100, A, MU

где

BA - количество используемых древесных пород (поле);

BB – группа наименований используемых древесных пород (периодическая группа);

B1 - порядковый номер наименования древесной породы (множественное поле);

B2 - наименование древесной породы (множественное поле);

B3 – назначение древесной породы (множественное поле);

B4 – количество способов распиловки (поле);

B5 – группа наименований способов распиловки (периодическая группа);

C1 - порядковый номер способа распиловки (множественное поле);

C2 – наименование способов распиловки (множественное поле);

B6 – количество сортов (поле);

B7 – группа сортов (периодическая группа);

D1 - порядковый номер сорта (множественное поле);

D2 – наименование сорта (множественное поле).

3.2. Уровень пользовательских приложений

3.2.1. Формирование списка активных элементов пользовательских приложений

Процедура формирования списка активных элементов пользовательских приложений осуществляется в следующем порядке:

1. Редактирование пользователем значений переменных ядра БД при помощи соответствующего пользовательского приложения.

2. Интерпретация и внесение изменений в файл параметров БД КИС (ядро модуля модификации).

3. Исполнение процедуры формирования списка активных элементов в момент открытия пользовательского приложения редактируемого программного модуля:

– Чтение значений переменных в ядре модуля модификации КИС.

– Интерпретация значений переменных.

– В зависимости от значений переменных - присвоение элементам открываемого приложения соответствующего статуса (активного, неактивного, видимого, невидимого и т.д.)

Каждая из процедур реализуется посредством исполнения соответствующего программного кода на языке Natural. Основы программирования в среде Natural изложены в [45, 183].

Программа вызывается встроенной событийно-ориентированной подпрограммой открываемого приложения, сигналом (событием) к исполнению которой послужит наступление определенного события.

Пример подпрограммы вызова:

perform sub01

где sub01 – название исполняемой программы.

Ниже показан пример вызываемой программы чтения значений переменных в ядре КИС (nas) и «выключения» (неактивные элементы) 2 элементов пользовательского приложения #tc-1 и #if-1, а также, изменения

текущего значения текстовой переменной на «Номер режима камерной сушки» при условии, что значение параметра aa в файле nas равно 1.

```
read (1) in nas
if aa = '1' then
assign #tc-1.visible := false
assign #tc-1.enabled := false
assign #if-1.visible := false
assign #if-1.enabled := false
move 'Номер режима камерной сушки' to #tc-2.string
end-if
end-read
```

В результате исполнения данного кода элементы пользовательского приложения #tc-1 и #if-1, будут недоступными, а строка элемента #tc-2 будет содержать значение «Номер режима камерной сушки».

На основе значений параметров изменяются как внутренние свойства элементов приложений (поля ввода, текстовые переменные, поля со списком), так и свойства самого приложения.

В дальнейшем, во всех последующих открываемых приложениях также производится редактирование списков активных элементов.

3.2.2. Формирование списка активных пользовательских приложений

Программный код интерпретации значений, указанных пользователем посредством электронной формы ввода и последующего формирования списка активных приложений КИС имеет следующий вид:

```
if #tb-2.checked = checked and #tb-2.enabled = true
move '1' to aa
move '1' to x3
end-if
if #tb-2.checked = unchecked or #tb-2.enabled = false
move '0' to aa
move '0' to x3
end-if
if #tb-1.checked = checked and #tb-1.enabled = true
move '1' to ap
move '1' to bp
```

```
move '1' to bq
move '1' to br
end-if
if #tb-1.checked = unchecked or #tb-1.enabled = false
move '0' to ap
move '0' to bp
move '0' to bq
move '0' to br
end-if
if #tb-4.checked = checked and #tb-4.enabled = true
move '1' to bs
end-if
if #tb-4.checked = unchecked or #tb-4.enabled = false
move '0' to bs
end-if
if #tb-5.checked = checked and #tb-5.enabled = true
move '1' to w1
end-if
if #tb-5.checked = unchecked or #tb-5.enabled = false
move '0' to w1
end-if
if #tb-9.checked = checked and #tb-9.enabled = true
move '1' to w2
end-if
if #tb-9.checked = unchecked or #tb-9.enabled = false
move '0' to w2
end-if
if #tb-12.checked = checked and #tb-12.enabled = true
move '1' to w3
end-if
if #tb-12.checked = unchecked or #tb-12.enabled = false
move '0' to w3
end-if
if #tb-14.checked = checked and #tb-14.enabled = true
move '1' to w4
end-if
if #tb-14.checked = unchecked or #tb-14.enabled = false
move '0' to w4
end-if
if #tb-20.checked = checked and #tb-20.enabled = true
move '1' to w5
end-if
if #tb-20.checked = unchecked or #tb-20.enabled = false
move '0' to w5
```

```
end-if
if (#tb-5.checked = checked and #tb-5.enabled = true)
or (#tb-9.checked = checked and #tb-9.enabled = true)
or (#tb-12.checked = checked and #tb-12.enabled = true)
or (#tb-14.checked = checked and #tb-14.enabled = true)
or (#tb-20.checked = checked and #tb-20.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
or (#tb-23.checked = checked and #tb-23.enabled = true)
or (#tb-24.checked = checked and #tb-24.enabled = true)
or (#tb-25.checked = checked and #tb-25.enabled = true)
move '1' to z1
move '1' to x1
end-if
if (#tb-5.checked = unchecked or #tb-5.enabled = false)
and (#tb-9.checked = unchecked or #tb-9.enabled = false)
and (#tb-12.checked = unchecked or #tb-12.enabled = false)
and (#tb-14.checked = unchecked or #tb-14.enabled = false)
and (#tb-20.checked = unchecked or #tb-20.enabled = false)
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
and (#tb-23.checked = unchecked or #tb-23.enabled = false)
and (#tb-24.checked = unchecked or #tb-24.enabled = false)
and (#tb-25.checked = unchecked or #tb-25.enabled = false)
move '0' to z1
move '0' to x1
end-if
if (#tb-5.checked = checked and #tb-5.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
move '1' to z2
move '1' to y2
move '1' to x2
end-if
if (#tb-5.checked = unchecked or #tb-5.enabled = false)
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
move '0' to z2
move '0' to y2
move '0' to x2
end-if
if (#tb-9.checked = checked and #tb-9.enabled = true)
or (#tb-14.checked = checked and #tb-14.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
move '1' to z3
end-if
if (#tb-9.checked = unchecked or #tb-9.enabled = false)
and (#tb-14.checked = unchecked or #tb-14.enabled = false)
```

```
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
move '0' to z3
end-if
if (#tb-5.checked = checked and #tb-5.enabled = true)
or (#tb-9.checked = checked and #tb-9.enabled = true)
or (#tb-14.checked = checked and #tb-14.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
move '1' to z4
move '1' to z8
end-if
if (#tb-5.checked = unchecked or #tb-5.enabled = false)
and (#tb-9.checked = unchecked or #tb-9.enabled = false)
and (#tb-14.checked = unchecked or #tb-14.enabled = false)
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
move '0' to z4
move '0' to z8
end-if
if (#tb-6.checked = checked and #tb-6.enabled = true)
or (#tb-9.checked = checked and #tb-9.enabled = true)
or (#tb-14.checked = checked and #tb-14.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
move '1' to z5
end-if
if (#tb-6.checked = unchecked or #tb-6.enabled = false)
and (#tb-9.checked = unchecked or #tb-9.enabled = false)
and (#tb-14.checked = unchecked or #tb-14.enabled = false)
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
move '0' to z5
end-if
if #tb-7.checked = checked and #tb-7.enabled = true
or (#tb-10.checked = checked and #tb-10.enabled = true)
or (#tb-15.checked = checked and #tb-15.enabled = true)
move '1' to z6
end-if
if #tb-7.checked = unchecked or #tb-7.enabled = false
and (#tb-10.checked = unchecked or #tb-10.enabled = false)
and (#tb-15.checked = unchecked or #tb-15.enabled = false)
move '0' to z6
end-if
if #tb-8.checked = checked and #tb-8.enabled = true
or (#tb-11.checked = checked and #tb-11.enabled = true)
or (#tb-16.checked = checked and #tb-16.enabled = true)
move '1' to z7
end-if
```

```
if #tb-8.checked = unchecked or #tb-8.enabled = false
and (#tb-11.checked = unchecked or #tb-11.enabled = false)
and (#tb-16.checked = unchecked or #tb-16.enabled = false)
move '0' to z7
end-if
if (#tb-12.checked = checked and #tb-12.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
move '1' to z9
end-if
if (#tb-12.checked = unchecked or #tb-12.enabled = false)
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
move '0' to z9
end-if
if #tb-13.checked = checked and #tb-13.enabled = true
move '1' to za
end-if
if #tb-13.checked = unchecked or #tb-13.enabled = false
move '0' to za
end-if
if #tb-23.checked = checked and #tb-23.enabled = true
move '1' to zb
end-if
if #tb-23.checked = unchecked or #tb-23.enabled = false
move '0' to zb
end-if
if #tb-24.checked = checked and #tb-24.enabled = true
move '1' to zc
end-if
if #tb-24.checked = unchecked or #tb-24.enabled = false
move '0' to zc
end-if
if #tb-20.checked = checked and #tb-20.enabled = true
or (#tb-22.checked = checked and #tb-22.enabled = true)
or (#tb-25.checked = checked and #tb-25.enabled = true)
move '1' to zd
end-if
if #tb-20.checked = unchecked or #tb-20.enabled = false
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
and (#tb-25.checked = unchecked or #tb-25.enabled = false)
move '0' to zd
end-if
if #tb-17.checked = checked and #tb-17.enabled = true
move '1' to y3
end-if
```

```
if #tb-17.checked = unchecked or #tb-17.enabled = false
move '0' to y3
end-if
if #tb-18.checked = checked and #tb-18.enabled = true
move '1' to y4
end-if
if #tb-18.checked = unchecked or #tb-18.enabled = false
move '0' to y4
end-if
if #tb-19.checked = checked and #tb-19.enabled = true
or (#tb-20.checked = checked and #tb-20.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
move '1' to y5
end-if
if #tb-19.checked = unchecked or #tb-19.enabled = false
and (#tb-20.checked = unchecked or #tb-20.enabled = false)
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
move '0' to y5
end-if
if (#tb-20.checked = checked and #tb-20.enabled = true)
or (#tb-22.checked = checked and #tb-22.enabled = true)
move '1' to x4
end-if
if (#tb-20.checked = unchecked or #tb-20.enabled = false)
and (#tb-22.checked = unchecked or #tb-22.enabled = false)
move '0' to x4
end-if
store record in kernel1
end
```

Запуск данного программного кода требует, чтобы область допустимых значений переменных (w1, x1, z1 и пр.) была определена. Программный код определения допустимых значений переменных приведенного выше программного кода, а также всех последующих программных кодов показан ниже:

```
DEFINE DATA LOCAL
1 KERNEL1 VIEW OF KERNEL1
2 AM (A1)
3 PM
3 AA (A1)
3 AZ
4 AP (A1)
4 A1
```

```
5 BP (A1)
5 B1
6 Z1 (A1)
6 Z2 (A1)
6 Z3 (A1)
6 Z4 (A1)
6 Z5 (A1)
6 Z6 (A1)
6 Z7 (A1)
6 Z8 (A1)
6 Z9 (A1)
6 ZA (A1)
6 ZB (A1)
6 ZC (A1)
6 ZD (A1)
5 BQ (A1)
5 B2
6 Y1 (A1)
6 Y2 (A1)
6 Y3 (A1)
6 Y4 (A1)
6 Y5 (A1)
6 Y6 (A1)
5 BR (A1)
5 B3
6 X1 (A1)
6 X2 (A1)
6 X3 (A1)
6 X4 (A1)
5 BS (A1)
5 B4
6 W1 (A1)
6 W2 (A1)
6 W3 (A1)
6 W4 (A1)
6 W5 (A1)
END-DEFINE
```

Пример формирования списка доступных пользователю приложений в соответствии со сформированным списком активных элементов меню управляющего приложения показан на рис. 3.5.

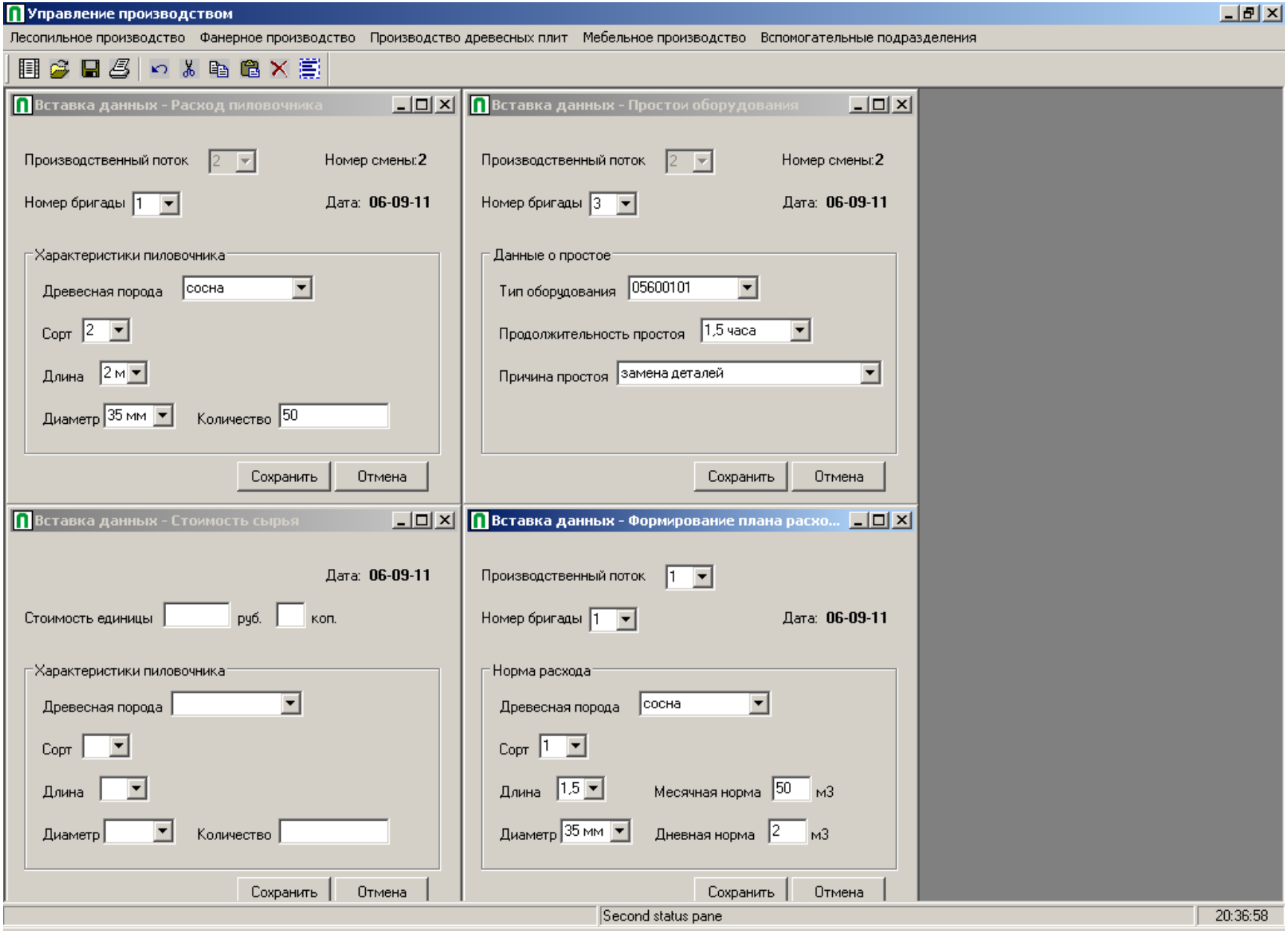


Рис. 3.5. Приложения, доступные пользователю на основе сформированного списка активных элементов

3.2.3. Автоматическая настройка активных пользовательских приложений

Идея автоматической настройки активных приложений КИС заключается в создании модели структуры КИС лесопромышленного предприятия в виде иерархической структуры однобайтовых полей, которая задается посредством назначения однобайтовым полям свойств взаимосвязанных полей, множественных полей, групп и периодических групп. Эта структура может существовать в виде одного или нескольких файлов БД. Непосредственно, настройка КИС, т.е. изменение перечня и порядка выполнения программных компонентов основных программных модулей КИС осуществляется путем редактирования значений, хранящихся в однобайтовых полях ядра модуля модификации (переменные ядра модуля модификации).

Чтобы представить взаимодействие описанной структуры с иными компонентами КИС рассмотрим структуру КИС лесопромышленного предприятия.

Каждый из программных модулей КИС лесопромышленного предприятия (перечень модулей дан в главе 1) может быть представлен как совокупность программ взаимодействия с ядром системы (программы-запросы, программы внесения данных и пр.), пользовательских приложений, программ основных расчетов и отчетных форм (рис. 1.1.). В рамках КИС программы-запросы, приложения расчетные программы и формы создаются как самостоятельные компоненты системы. Компоненты системы могут быть:

– стандартизированные (обязательные для использования программными модулями КИС);

– специализированные (используемые выборочно, в рамках специализированных программных модулей или в рамках основных программных модулей, модифицированных с учетом особенностей предприятия).

Программные модули КИС – это совокупность стандартизированных и специализированных компонентов системы.

Для обеспечения возможности автоматической настройки приложений КИС, необходимо каждый из компонентов КИС наделить свойством исполнения (запуска) на основе значений переменных ядра модуля модификации.

Для осуществления модификации КИС лесопромышленного предприятия предусматривается редактирование 2 групп файлов параметров:

– Файлы значений переменных элементов записей. Значения переменных изменяются в зависимости от технологических особенностей предприятия.

– Файлы значений переменных, на основе которых осуществляется формирование списка активных элементов приложений и перечня программ взаимодействия, программ основных расчетов и отчетных форм, используемых программными модулями КИС. Значения этих параметров изменяются в соответствии с отраслевой и внутриорганизационной спецификой лесопромышленного предприятия.

Таким образом, структура ядра модуля модификации КИС лесопромышленного предприятия состоит из файлов значений переменных элементов записей и файлов значений переменных модификации КИС.

При запуске компонентов КИС (в т.ч. приложений) на основе значений полей ядра модуля модификации событийно-ориентированными подпрограммами формируется список активных, т.е. отображаемых и изменяемых элементов (таких, как поле ввода или переключатель), а также свойства этих элементов. Активные элементы приложений событийно-ориентированными подпрограммами связываются с приложениями доступа и манипулирования данными, приложениями, осуществляющие управление программами взаимодействия с БД КИС и программами основных расчетов.

Значения некоторых переменных не подлежат редактированию со стороны пользователя КИС. Перечень таких переменных определяется поставщиком КИС, в зависимости от заказанного пользователем модульного состава КИС.

Для хранения значений этих переменных формируется отдельный файл в БД системы. Доступ к полям, содержащим значения переменных, не подлежащих редакции пользователями КИС запрещается. Редактирование значений остальных переменных осуществляется пользователем посредством приложений модификации КИС. Доступ к таким приложениям защищается администраторским паролем.

Осуществление автоматической настройки приложений КИС лесопромышленного предприятия модулем модификации может быть представлено в виде схемы (рис. 3.6.)

На рис. 3.6. введены следующие условные обозначения:

1 – на основе пароля доступа формирование списка параметров (полей), значения которых могут редактироваться пользователем;

2 – изменение значений идентификаторов элементов записей и внесение изменений в файл БД КИС;

3 – изменение значений параметров формирования активных элементов и внесение изменений в файл БД КИС;

4 – формирование области значений идентификаторов записи, как самостоятельного компонента программного модуля;

5 - формирование области значений параметров, как самостоятельного компонента программного модуля на основе значений параметров;

6 – формирование списка специализированных программ взаимодействия с БД КИС, используемых в рамках программных модулей КИС на основе области значений идентификаторов элементов записи и области значений параметров;

7 - формирование списка специализированных пользовательских приложений и активных элементов приложений, используемых в рамках программных модулей КИС на основе области значений идентификаторов элементов записи и области значений параметров;

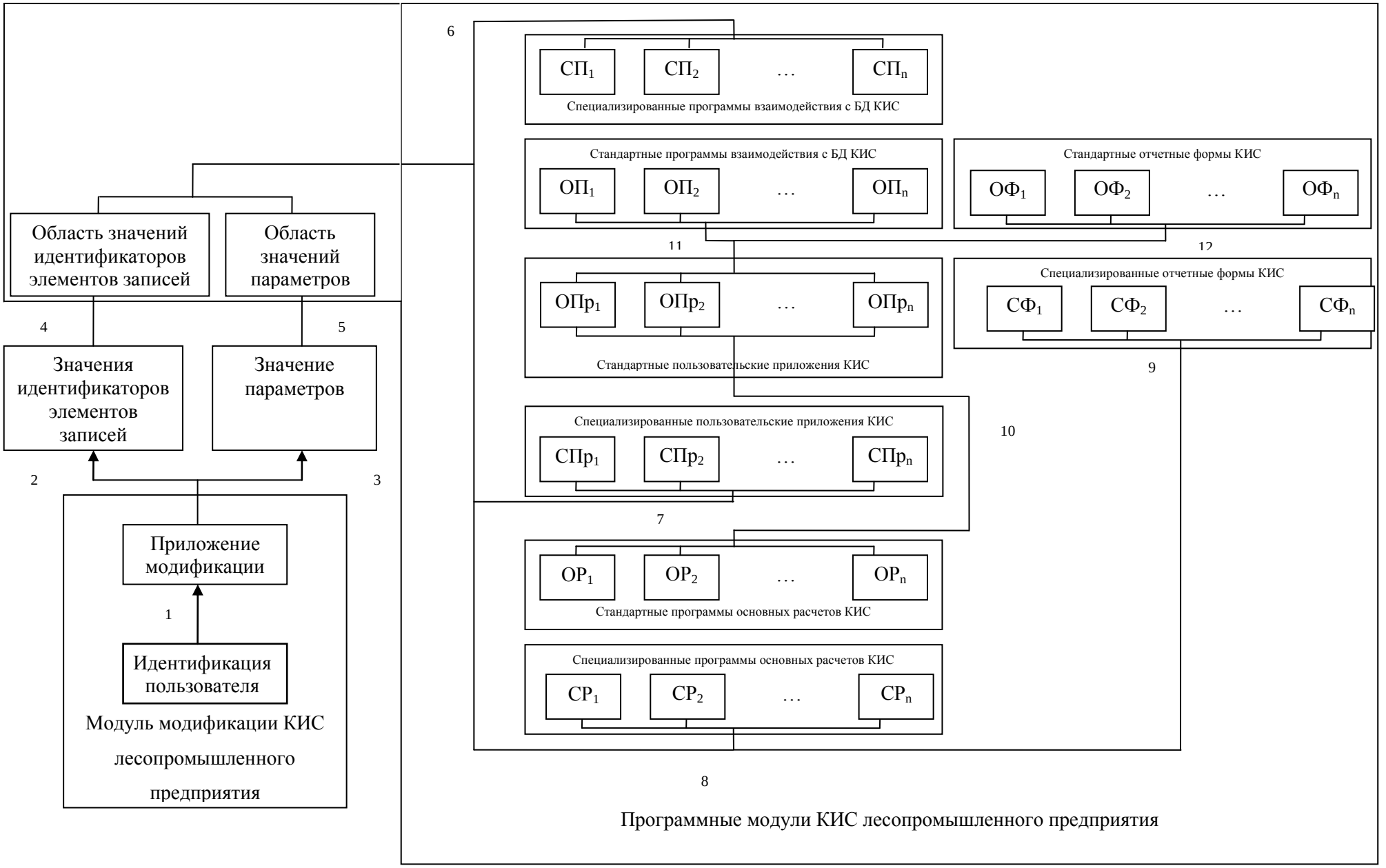


Рис. 3.6. Схема процесса автоматической настройки приложений КИС лесопромышленного предприятия

8 - формирование списка специализированных программ основных расчетов, используемых в рамках программных модулей КИС на основе области значений идентификаторов элементов записи и области значений параметров;

9 - формирование списка специализированных отчетных форм, используемых в рамках программных модулей КИС на основе области значений идентификаторов элементов записи и области значений параметров;

10, 11, 12 – формирование списков стандартных программ взаимодействия с БД КИС, программ основных расчетов и отчетных форм происходит на основе встроенных процедур пользовательских приложений. Перечень стандартных пользовательских приложений является неизменным для любых программных модулей КИС лесопромышленного предприятия.

3.3. Уровень расчетных программ КИС

3.3.1. Оптимизация запросов

При оптимизации на уровне расчетных программ ведущее место занимает организация запросов к БД. Нерационально составленный запрос замедляет работу системы, перегружает рабочие станции, отвлекает ресурсы серверов на излишнюю обработку данных.

Для обеспечения эффективности работы расчетных программ запросы должны обладать следующими свойствами:

- возможность выбора пользователем критериев запроса;
- использование оптимального количества критериев запроса;
- оптимальная логическая структура запроса.

Для обеспечения возможности проведения запросов на основе критериев, задаваемых пользователем, указывается значение соответствующей переменной (поле ZD в примере в табл. 3.3.)

На основе значения этой переменной в приложении управления программным модулем КИС (поле X3 табл.3.3.) активизируются элементы,

связанные средствами событийно-ориентированного программирования с приложением выбора критериев поиска (поле X4 табл.3.3.).

В целях обеспечения производительности рекомендуется организовывать запросы данных на основе перечня не более чем 8-10 критериев. Часть перечня формируется пользователем (3-5 критериев), и часть - являются фиксированными (3-5 критериев дата, номер смены, номер производственного подразделения и т.д.). Значения критериев в обоих случаях задаются пользователем.

Процесс осуществления запроса, дальнейшей обработки данных, расчетов и представления результатов в виде отчетной формы представлен на рис.3.7.

На рис. 3.7. использованы следующие условные обозначения:

1 – передача результата выбора критериев поиска (экспорт списка критериев поиска).

2 – передача значений критериев поиска.

3 – передача интерпретированных значений критериев поиска.

4 – передача во временный файл результатов поиска на основе значений критериев выбранных пользователем.

5 – передача во временный файл результатов поиска на основе значений критериев выбранных пользователем.

6 – запуск поиска на основе фиксированных критериев поиска.

7 – передача в рабочий файл результатов поиска на основе значений выбранных и фиксированных критериев запроса и значений параметров файла БД модуля модификации.

8 – передача результатов поиска на основе значений выбранных и фиксированных критериев запроса и значений параметров файла БД модуля модификации для проведения расчетов.

9 – передача результатов расчетов.

10 – передача интерпретированных значений параметров и результатов расчета.

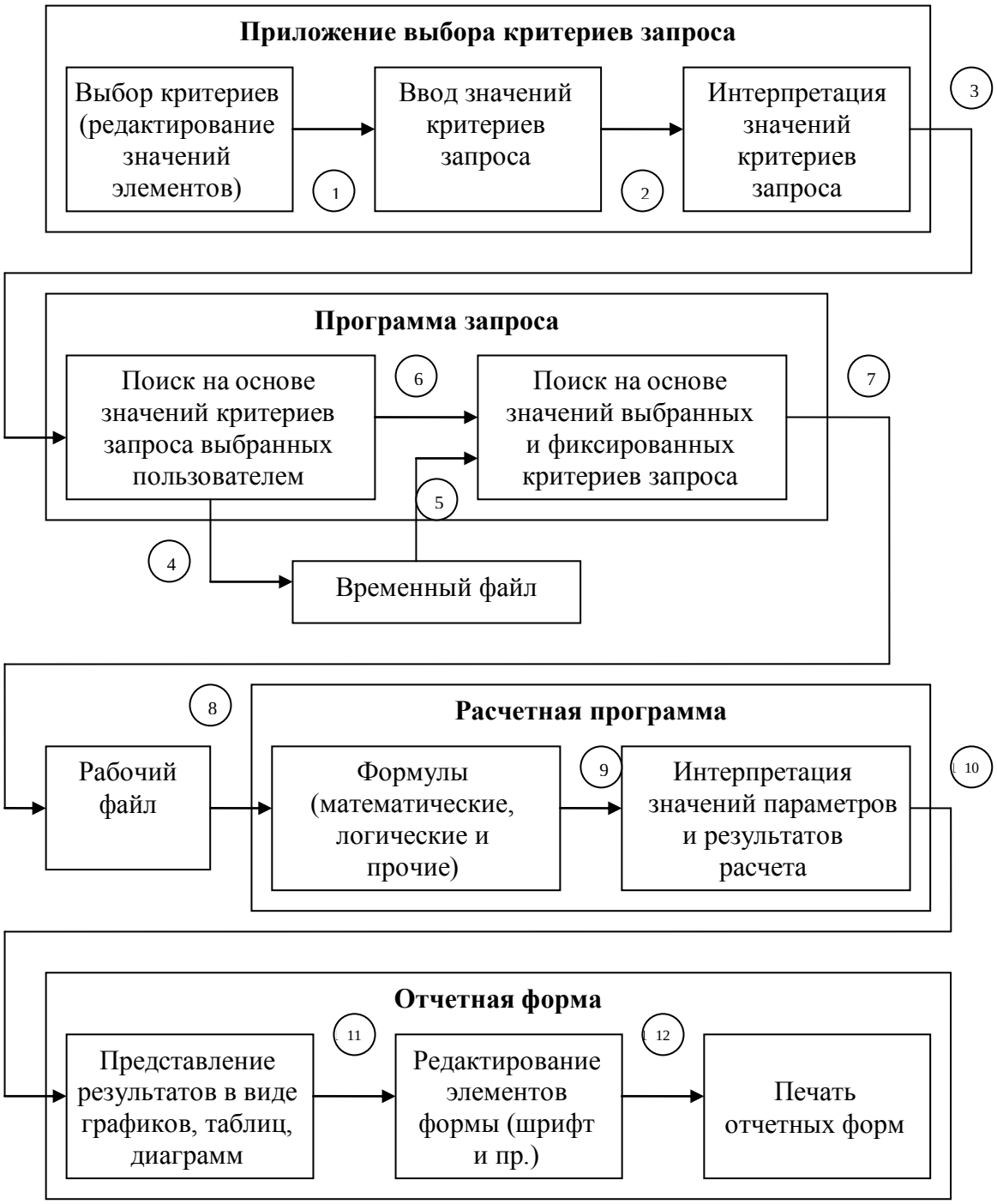


Рис. 3.7. Схема процесса запроса, дальнейшей обработки и расчетов, и представление результатов в виде отчетной формы

11 – представление элементов формы (таблиц, диаграмм, графиков) для редактирования.

12 – подача отчетной формы на печать.

Приложение выбора критериев запроса предлагает пользователю:

– выбор критериев поиска и ввод значений выбранных критериев

– выбор фиксированных критериев при помощи редактирования значений элементов приложения (полей со списком, альтернативных списков и пр.).

После ввода значений критериев включается алгоритм интерпретации введенных значений. Интерпретированные значения являются базой для поиска.

Ниже приведен код программы интерпретации значений 3 критериев, выбираемых пользователем при помощи 6 элементов приложения выбора критериев (#tb-1, #tb-2, #tb-3 – индикаторы использования критериев; #sb-1, #sb-2, #sb-3 – поля ввода значений соответствующих критериев) в рамках программного блока управления лесопильным производством:

```
if #tb-1.checked = checked then
move '1' to #b1
move #sb-1.string to #a1
end-if
if #tb-1.checked = unchecked then
move '0' to #b1
end-if
if #tb-2.checked = checked then
move '1' to #b2
move #sb-2.string to #a2
end-if
if #tb-2.checked = unchecked then
move '0' to #b2
end-if
if #tb-3.checked = checked then
move '1' to #b3
move #sb-3.string to #a3
end-if
if #tb-3.checked = unchecked then
move '0' to #b3
end-if
```

Показанный пример присваивает значение «1» переменным #b1, #b2, #b3 и значения полей ввода #sb-1, #sb-2, #sb-3 переменным #a1, #a2, #a3, в случае включения индикаторов использования критериев поиска #tb-1, #tb-2, #tb-3. Если какие-либо из индикаторов #tb-1, #tb-2, #tb-3 не включены, соответствующим переменным #b1, #b2, #b3 присваивается значение «0».

Следующий программный код осуществляет поиск данных в файле БД uchmat на основе значений переменных #b1, #b2, #b3 и #a1, #a2, #a3 и сохранение результатов поиска во временном файле «first»:

```
if #b1 = '1' and #b2 = '0' and #b3 = '0' then find uchmat with a1 = #a1
retain as ‘first’
end-find
end-if
if #b1 = '0' and #b2 = '1' and #b3 = '0' then find uchmat with a2 = #a2
retain as ‘first’
END-find
end-if
if #b1 = '0' and #b2 = '0' and #b3 = '1' then find uchmat with a3 = #a3
retain as ‘first’
END-find
end-if
if #b1 = '1' and #b2 = '1' and #b3 = '0' then find uchmat with a1 = #a1 and a2 = #a2
retain as ‘first’
END-find
end-if
if #b1 = '1' and #b2 = '0' and #b3 = '1' then find uchmat with a1 = #a1 and a3 = #a3
retain as ‘first’
END-find
end-if
if #b1 = '0' and #b2 = '1' and #b3 = '1' then find uchmat with a3 = #a3 and a2 = #a2
retain as ‘first’
END-find
end-if
if #b1 = '1' and #b2 = '1' and #b3 = '1' then find uchmat with a3 = #a3 and a2 = #a2
and a1 = #a1
retain as ‘first’
END-find
end-if
```

Фиксированные критерии поиска являются обязательными для ввода. Поэтому наличие индикаторов включения данных критериев (присвоение значения «0» для соответствующих переменных ядра модуля) не является необходимым. Показанный ниже код осуществляет поиск данных на основе введенных значений фиксированных критериев (поля ввода значений критериев #sb-4, #sb-5, #sb-6) и результатов поиска на основе критериев, выбираемых пользователем (файл «first»).

move #sb-4.string to #a4
move #sb-5.string to #a5
move #sb-6.string to #a6
find uchmat with ‘first’ and a4 = #a4 and a5 = #a5 and a6 = #a6

Результаты поиска заносятся в рабочий файл. Пример программного кода создания рабочего файла, и сохранения результатов поиска показан ниже:

DEFINE WORK FILE n 'путь'
WRITE WORK FILE n VARIABLE
x1 x2 x3 x4 x5
CLOSE WORK FILE n
где n – порядковый номер создаваемого рабочего файла;

'путь' – месторасположение файла на физическом носителе (задается автоматически);

x1, x2, x3, x4, x5 – названия переменных, значения которых сохраняются в рабочий файл (в данном случае это названия полей БД, по значениям которой осуществлялся поиск, а также значения параметров файла БД модуля модификации).

Над значениями переменных, сохраненных в рабочие файлы, производятся основные вычисления. Элементы отчетных форм редактируются на основе значений параметров рабочих файлов аналогичным образом.

3.3.2. Использование различных типов переменных

В рамках СУБД ADABAS и Natural используется большое количество типов переменных, которые отличаются:

– по формату данных;

– по обработке хранимых значений;

– по возможности проведения запросов к хранимым значениям;

– по возможности проведения математических и логических операций с хранимыми значениями и т.д.

В целях оптимизации программных кодов необходимо предусматривать оптимальное использование различных типов переменных.

В СУБД ADABAS в целях запросы данных могут осуществляться только к поисковым полям (дескрипторам) различных типов. Любое поле при

определении FDT может быть определено дескриптором. Для поля, определенного дескриптором будет сделан вход в инвертированном списке ассоциатора, который позволит этому полю: использоваться в выражении поиска и в качестве ключа сортировки в команде FIND, управлять последовательным логическим чтением или использоваться для связи файла.

Используются следующие типы дескрипторов:

- DE – дескриптор;
- SB – субдескриптор;
- SP – супердескриптор;
- HY – гипердескриптор;
- PH – фонетический дескриптор.

Субдескриптор - это дескриптор, созданный из части элементарного поля. Элементарное поле может быть или не быть дескриптором непосредственно. Субдескриптор может также использоваться, как субполе; то есть, он может быть определен в форматном буфере для определения формата выходных записей.

Субдескриптор должен быть определен со следующими параметрами:

Имя - имя субдескриптора. Правила присвоения имени субдескриптору, идентичны тем, которые используются для имени поля ADABAS.

UQ (опция Unique) - указывает, что субдескриптор должен быть определен как уникальный.

Исходное поле - имя исходного поля, из которого будет получен субдескриптор.

Начало - относительная позиция байта в пределах исходного поля, где начинается описание субдескриптора.

Конец - относительная позиция байта в пределах исходного поля, где кончается описание субдескриптора.

Подсчет параметров начало и конец делается слева направо, начиная с 1 для алфавитно-цифровых полей, и справа налево, начиная с 1 для цифровых и двоичных полей. Если исходное поле определено с форматом P, знак

результирующего значения субдескриптора будет взят из 4 битов младшего разряда байта младшего разряда (т. е. 1 байт).

Исходное поле может быть:

– дескриптором;

– элементарным полем;

– множественным полем (но не конкретная реализация множественного поля);

– входить в состав периодической группы (но не конкретная реализация периодической труппы).

Исходное поле не может быть:

– периодической группой, субдескриптором, супердескриптором или фонетическим дескриптором;

– G формата (с плавающей точкой).

Супердескриптор - это дескриптор, созданный из нескольких полей, частей полей или их комбинации. Каждое поле-первоисточник (или часть поля) используемое для определения супердескриптора, называется исходным. Супердескриптор может быть определен с использованием от 2 до 20 исходных полей или частей поля. Супердескриптор может также быть определен, как уникальный дескриптор. Супердескриптор может также использоваться, как суперполе; то есть, он может быть определен в форматном буфере для определения формата выходной записи.

Субдескриптор должен быть определен со следующими параметрами:

Имя – имя супердескриптора. Правила присвоения имени супердескриптору, идентичны тем, которые используются для имени поля ADABAS.

UQ (опция Unique) - указывает, что супердескриптора должен быть определен как уникальный (см. описание опции UQ на странице 50).

Исходное поле - имя исходного поля, из которого будет получен элемент супердескриптора; может быть определено до 20 исходных полей.

Начало - относительная позиция байта в пределах исходного поля, где начинается описание супердескриптора.

Конец - относительная позиция байта в пределах исходного поля, где заканчивается описание супердескриптора.

Подсчет параметров начало и конец делается слева направо, начиная с 1 для алфавитно-цифровых полей, и справа налево, начиная с 1 для цифровых и двоичных полей. Для любого исходного поля кроме тех, которые определены как "FI", значения начала и конца имеют силу в пределах разрешенного диапазона для типа данных исходного поля.

Исходное поле может быть:

– элементарным или максимумом одним множественным полем (но не определенного значения множественного поля);

– в пределах периодической группы (но не определенная реализация);

– дескриптором.

Исходное поле не может быть:

– супер-, суб- или фонетическим дескриптором;

– G формата (с плавающей точкой);

– поле опции NC, если другое исходное поле - поле опции NU;

– длинное алфавитно-цифровое (LA) поле.

Полная длина любого значения супердескриптора не может превышать 253 байта (алфавитно-цифровые) или 126 (двоичных) байтов. Супердескриптор имеет алфавитно-цифровой формат, если какой-нибудь элемент супердескриптора получен из алфавитно-цифрового исходного поля; иначе, супердескриптор имеет двоичный формат. Все супердескрипторы двоичного формата обрабатываются, как числа без знака.

Гипердескриптор позволяет сгенерировать значения дескриптора на основании алгоритма, обеспеченного пользователем.

Значения основаны на алгоритмах, закодированных в специальных пользовательских выходах гипердескриптора (от HEX01 до HEX31). Каждому

гипердескриптору должен быть назначен пользовательский выход, и отдельный пользовательский выход может обрабатывать множество гипердескрипторов (рис. 3.8.).

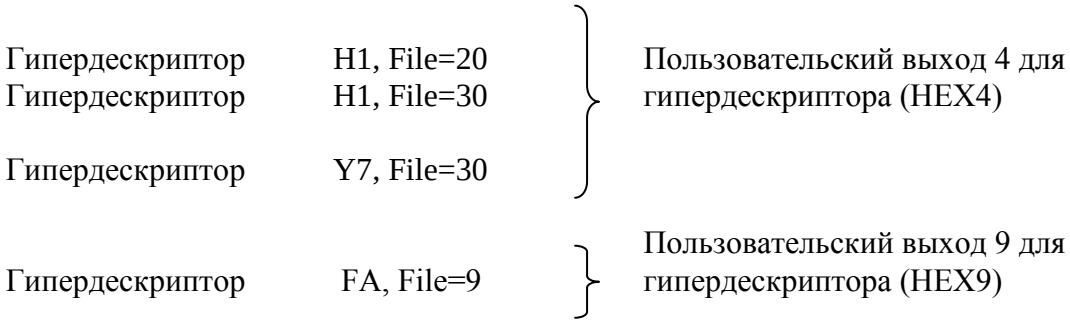


Рис. 3.8. Гипердескриптор и пользовательские выходы

Входные параметры необходимые для пользовательского выхода:

– имя гипердескриптора;

– номер файла;

– адреса полей, взятых из записи памяти данных, вместе с именем поля и PE индексом (если применяется). Эти адреса указывают на сжатые значения полей.

Пользовательский выход должен возвратить значение дескриптора (DVT) в сжатом формате. Может быть не возвращено никакого значения или одно или большее количество значений в зависимости от опций (PE, MU), определенных для гипердескриптора.

Исходный ISN, назначенный для входного значения, может быть изменен.

При использовании гипердескрипторов следует учитывать следующие примечания:

– Проектировщик определяет формат, длину и опции гипердескриптора. Они не могут быть взяты от исходного поля, определенного в параметре HY.

– Поиск, использующий значение гипердескриптора, выполняется тем же способом, что и для стандартных дескрипторов.

– Проектировщик ответствен за создание корректных значений гипердескриптора. Нет никакого стандартного пути проверки значений

гипердескриптора на целостность в памяти данных. Пользователь должен установить правила описания каждого значения и проверки корректности значения.

– Если формат гипердескриптора упакованный или распакованный, ADABAS проверяет возвращенные значения на правильность. Знак полбайта для упакованного значения может содержать A, C, E, F (положительный) или B, D (отрицательный). ADABAS преобразует знак в F или D.

– Если файл содержит больше одного гипердескриптора, назначенные выходы вызываются в алфавитном порядке имен гипердескрипторов.

Гипердескриптор должен быть определен со следующими параметрами:

Номер - номер пользовательского выхода, который будет назначен для гипердескриптора. Ядро ADABAS использует этот номер для определения пользовательского выхода гипердескриптора, который будет вызван.

Имя - имя, которое используется для гипердескриптора. Правила присвоения имени гипердескриптору идентичны тем, которые используются для имени поля ADABAS.

Длина - длина гипердескриптора по умолчанию.

Формат – формат гипердескриптора:

Формат	Максимальная длина
Алфавитно-цифровой (A)	253 байта
Двоичный (B)	126 байтов
С фиксированной точкой (F)	4 байта (всегда 2 или 4 байта)
С плавающей точкой (G)	8 байтов (всегда 4 или 8 байтов)
Упакованный десятичный (P)	15 байтов
Распакованный десятичный (U)	29 байтов

Опция - опция, которая будет назначена гипердескриптору.

Исходное поле - имя исходного поля. Гипердескриптор может иметь от 1 до 20 исходных полей. Имена полей и адреса передаются пользовательскому выходу.

Если исходное поле определено с опцией NU, никакие входы не генерируются в инвертированном списке гипердескриптора для записей, содержащих нулевые значения поля. Это действительно не зависит от присутствия или отсутствия значений для других элементов гипердескриптора.

Для лучшего понимания процедуры определения гипердескриптора ниже приведен пример описания гипердескриптора.

Для этого примера используются следующее описание (табл. 3.4.):

Пример описания гипердескриптора

Таблица 3.4.

Описание определенных полей	Значения полей
1,LN,20,A,DE,NU	Фамилия
1,FN,20,A,MU,NU	Имя
1,ID,4,B,NU	Идентификатор
1,AG,3,U	Возраст
1,AD,PE	Адрес
2,CI,20,A,NU	Город
2,ST,20,A,NU	Улица
1,FA,PE	Родственники
2,NR,20,A,NU	Фамилия родственника
2,FR,20,A,MU,NU	Имя родственника

Описание гипердескриптора: HYPDE=’2,NH,60,A,MU,NU=LN,FN,FR’

Гипердескриптору назначен пользовательский выход 2 и имя гипердескриптора - HN. Длина гипердескриптора - 60, формат - алфавитно-цифровой и множественное (MU) поле с подавлением нулей (NU).

Фонетический дескриптор позволяет проводить запросы, при которых все возвращенные записи будут содержать подобные фонетические значения (иметь аналогичное произношение). Фонетическое значение дескриптора основано на первых 20 байтах значения поля. Рассматриваются только алфавитные значения; числовые значения, специальные символы и пробелы игнорируются. Нижний и верхний регистры алфавитно-цифровых символов внутренне идентичны.

Фонетический дескриптор должен быть определен со следующими параметрами:

Имя - имя, которое будет использоваться для фонетического дескриптора. Правила присвоения имени фонетическому дескриптору, идентичны тем, которые используются для имени поля ADABAS.

Поле - имя поля, которое будет транскрибировано фонетически.

Поле должно быть:

– элементарным или множественным полем;

– определенно в алфавитно-цифровом формате.

Поле может быть дескриптором.

Поле не может:

– быть субдескриптором, супердескриптором или гипердескриптором;

– входить в состав периодической группы;

– использоваться, как исходное поле, для более, чем одного фонетического дескриптора.

3.3.3. Формирование списка активных расчетных программ

Этапы формирования списка активных расчетных программ КИС следующие:

1. Создание связи активных элементов пользовательских приложений с запуском различных компонентов КИС на основе значений переменных ядра модуля модификации.

2. Редактирование пользователем значений переменных ядра модуля модификации, интерпретация и внесение изменений в ядро модуля модификации.

3. В процессе работы с программными приложениями, в момент наступления определенных событий, связанных с активными элементами приложений - чтение значений переменных ядра модуля модификации, интерпретация и вызов соответствующих программных компонентов КИС.

Связь активных расчетных программ с запуском программных компонентов КИС на основе значений переменных ядра осуществляется при помощи средств событийно-ориентированного программирования [45]. Фактически, она подразумевает исполнение программного кода, запускающего определенный компонент КИС в зависимости от значений тех или иных переменных ядра модуля модификации.

Примером осуществления связи активных программ с запуском программных компонентов КИС служит следующий программный код:

```
DECIDE ON FIRST *EVENT
      VALUE 'CLICK'
        OPTIONS 2 CLICK
        if bq = '1' then run sub1
        end-if
        if bq = '0' then run sub2
        end-if
        OPTIONS 3
    NONE
      PERFORM #DLG$HANDLER$DEFAULT
    END-DECIDE
```

Данный программный код обеспечивает запуск подпрограммы «sub1» в случае, когда значение переменной bq (индикатор включения расчетных программ) равно «1»; и запуск подпрограммы «sub2» в случае, когда значение переменной bq равно «0».

Основными событиями, служащими сигналом для запуска программных компонентов КИС являются:

- редактирование активного элемента;
- ввод значения активного элемента;
- переключение на другой активный элемент;
- щелчок мыши (для кнопок, альтернативных списков и пр.);
- двойной щелчок мыши;
- открытие приложения;
- закрытие приложения;
- прочие.

Показанный ниже программный код после щелчка мыши по кнопке #pb-1 выполняет запуск программного компонента «sub1» при значении переменной bq равной «1», и возвращающей пользователю сообщение о недоступности компонента, если значение переменной bq не равно «1»:

```
DEFINE DATA
  PARAMETER
    01 #DLG$PARENT HANDLE OF GUI BY VALUE
    LOCAL
      01 #DLG$WINDOW HANDLE OF WINDOW
      01 #PB-1 HANDLE OF PUSHBUTTON
END-DEFINE
VALUE #PB-1
    DECIDE ON FIRST *EVENT
      VALUE 'CLICK'
        OPTIONS 2 CLICK
        read (1) in nas
        if bq = '1' then run sub1
        end-if
        IF NOT (bq = '1')
        assign text = 'Данный программный компонент не активен'
        assign title = 'Изменение данных'
        assign stil = 'SO'
        open dialog ngu-messagebox
        using #dlg$window
        with #msg-box
        end-if
        end-read
        OPTIONS 3
    NONE
      PERFORM #DLG$HANDLER$DEFAULT
    END-DECIDE
```

3.3.4. Использование системы индикаторов расчетными программами

Система индикаторов КИС лесопромышленного предприятия представляет собой многоуровневую структуру однобайтовых полей и отражает структуру программных модулей КИС. Методы создания многоуровневых структур СУБД ADABAS рассмотрены в [45, 182].

Система индикаторов КИС служит для создания связей между расчетными программами и компонентами КИС (приложениями, формами, модулями и пр.), по отношению к которым эти программы исполняются.

В таблице 3.5. показаны индикаторы на основе, которых формируются значения компонентов посменного учета расхода пиловочного сырья производственными подразделениями предприятия (соответствующие элементам табл.3.3.).

Описание индикаторов посменного учета пиловочного сырья модуля управления лесопильным производством

Таблица 3.5.

№ п/п	Название индикатора	Номера активизируемых индикаторов	Соответствующие элементы таблицы FDT (табл. 3.1.)
1	Модуль управления лесопильным производством	2	2 , AA, 1, A
2	Посменный учет пиловочного сырья	3; 20	3 , AP, 1, A
3	Отчетные формы	4; 8; 11;13; 19	4 , BS, 1, A
4	Ведомость выполнения плана раскроя сырья	5; 6; 7	5 , W1, 1, A 5 , Z1, 1, A 5 , Z2, 1, A 5 , Z4, 1, A 5 , Z8, 1, A 5 , Y2, 1, A 5 , X1, 1, A 5 , X2, 1, A
5	Учет выполнения плана раскроя сырья по заданному сорту и древесной породе	-	5 , Z5, 1, A
6	Учет выполнения плана раскроя сырья по заданному сорту	-	5 , Z6, 1, A

продолжение табл. 3.5.

7	Учет выполнения плана раскроя сырья по заданной длине и диаметру	-	5 , Z7, 1, A
8	Ведомость объемов распиленного пиловочника	9; 10	5 , W2, 1, A 5 , Z1, 1, A 5 , Z3, 1, A 5 , Z4, 1, A 5 , Z5, 1, A 5 , Z8, 1, A 5 , X1, 1, A
9	Учет количества израсходованного сырья по заданному сорту	-	5 , Z6, 1, A
10	Учет общего количества израсходованного сырья по заданной длине и диаметру	-	5 , Z7, 1, A
11	Ведомость простоя оборудования	12	5 , W3, 1, A 5 , Z1, 1, A 5 , Z9, 1, A 5 , X1, 1, A
12	Учет продолжительности и причин простоя на всех производственных стадиях на определенную дату и за промежуток времени	-	5 , ZA, 1, A
13	Ведомость стоимости пиловочного сырья	14; 15; 16; 17; 18	5 , W4, 1, A 5 , Z1, 1, A 5 , Z3, 1, A 5 , Z4, 1, A 5 , Z5, 1, A 5 , Z8, 1, A 5 , Y1, 1, A 5 , X1, 1, A
14	Учет стоимости израсходованного сырья по заданному сорту	-	5 , Z6, 1, A
15	Учет стоимости израсходованного сырья по заданной длине и диаметру	-	5 , Z7, 1, A
16	Учет стоимости сырья всех видов, израсходованного производственным подразделением за месяц	-	5 , Y3, 1, A

продолжение табл. 3.5.

17	Учет стоимости сырья всех видов, израсходованного на производственных стадиях за месяц	-	5 , Y4, 1, A
18	Учет стоимости израсходованного сырья за единицу времени по заданному виду сырья, сорту и размерам	-	5 , Y5, 1, A
19	Отчетная форма количества израсходованного сырья по задаваемым критериям	-	5 , W5, 1, A 5 , Z1, 1, A 5 , ZD, 1, A 5 , Y5, 1, A 5 , X1, 1, A 5 , X4, 1, A
20	Дополнительные параметры посменного учета пиловочного сырья	21; 22; 23; 24	-
21	Применение стандартных параметров посменного учета пиловочного сырья	-	5 , Z1, 1, A 5 , Z2, 1, A 5 , Z3, 1, A 5 , Z4, 1, A 5 , Z5, 1, A 5 , Z8, 1, A 5 , Z9, 1, A 5 , Y1, 1, A 5 , Y2, 1, A 5 , Y5, 1, A 5 , X1, 1, A 5 , X2, 1, A 5 , X4, 1, A 5 , ZD, 1, A
22	Учет количества сырья всех видов, израсходованного производственным подразделением за месяц	-	5 , ZB, 1, A 5 , Z1, 1, A 5 , X1, 1, A
23	Учет количества сырья всех видов, израсходованного на производственных стадиях за месяц	-	5 , ZC, 1, A 5 , Z1, 1, A 5 , X1, 1, A
24	Учет количества использованного сырья по критериям, выбираемым и задаваемым пользователем	-	5 , ZD, 1, A 5 , Z1, 1, A 5 , X1, 1, A

3.4. Уровень отчетных форм

3.4.1. Получение отчетных форм на основе задаваемых критериев

Пример генерации отчетной формы на основе критериев, задаваемых пользователем показан на рис. 3.9.

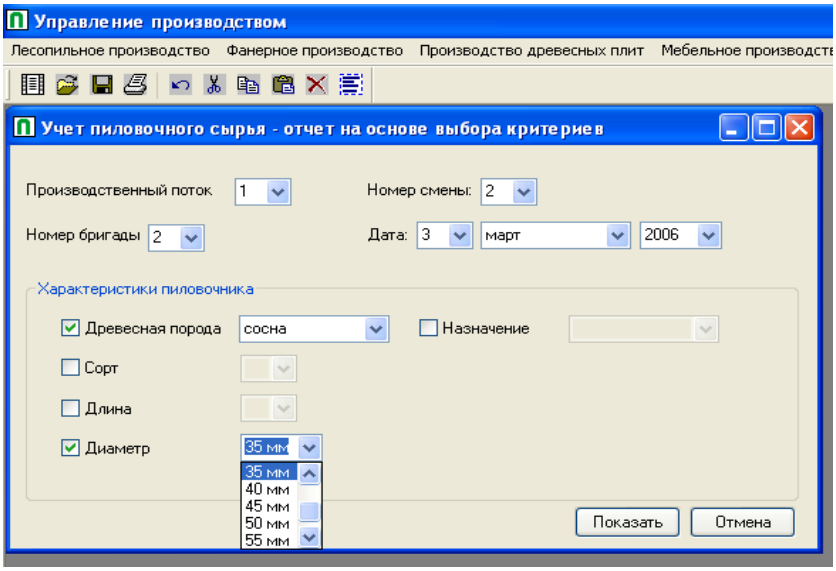
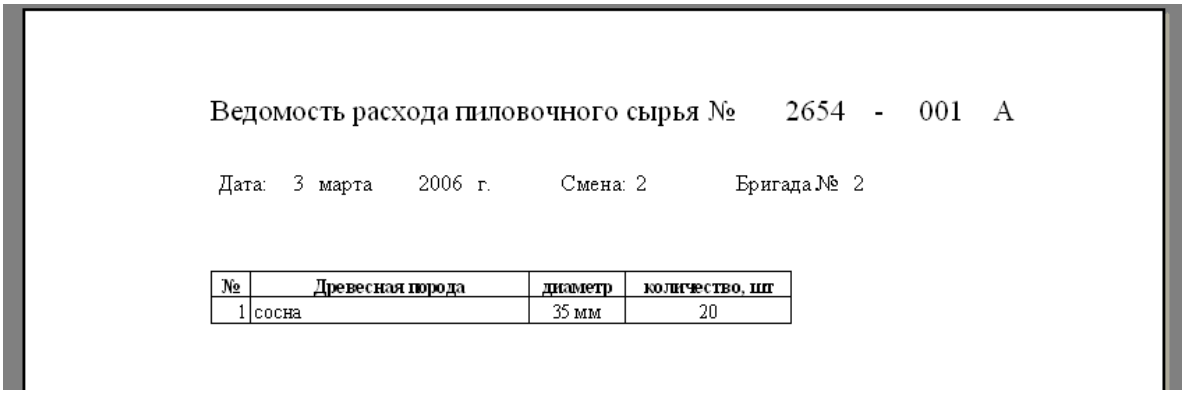


Рис. 3.9. Генерация отчетных форм на основе критериев, задаваемых пользователем

На основе значений критериев, заданных пользователем выполняется запрос на выборку данных. Более подробно технология осуществления запроса на выборку данных рассмотрена в п.3.3. и в [45, 182-197]. По желанию пользователя результат может быть выведен на экран и распечатан в виде отчетной ведомости (рис. 3.10.).



Ведомость расхода пиловочного сырья № 2654 - 001 А

Дата: 3 марта 2006 г. Смена: 2 Бригада № 2

№	Древесная порода	диаметр	количество, шт
1	сосна	35 мм	20

Рис. 3.10. Вывод отчетной формы (ведомости) сгенерированной на основе критериев пользователя

3.4.2. Формирование списка активных отчетных форм

Процедура формирования списка активных отчетных форм показана на рис. 3.11 и 3.12. Посредством управляющего меню приложения задается перечень доступных пользователю отчетных форм (рис. 3.11. и 3.12.).

Программный код формирования списка активных отчетных форм имеет следующий вид:

```
read (1) in kernel1
if ap = '1' assign #MITEM-4.enabled := true
end-if
if ap = '0' assign #MITEM-4.enabled := false
end-if
if bs = '1' assign #MITEM-30.enabled := true
end-if
if bs = '0' assign #MITEM-30.enabled := false
end-if
if w1 = '1' assign #MITEM-33.enabled := true
end-if
if w1 = '0' assign #MITEM-33.enabled := false
end-if
if w2 = '1' assign #MITEM-34.enabled := true
end-if
if w2 = '0' assign #MITEM-34.enabled := false
end-if
if w3 = '1' assign #MITEM-35.enabled := true
end-if
if w3 = '0' assign #MITEM-35.enabled := false
end-if
if w4 = '1' assign #MITEM-36.enabled := true
end-if
if w4 = '0' assign #MITEM-36.enabled := false
end-if
if w5 = '1' assign #MITEM-37.enabled := true
end-if
if w5 = '0' assign #MITEM-37.enabled := false
end-if
if zb = '1' assign #MITEM-46.enabled := true
end-if
if zb = '0' assign #MITEM-46.enabled := false
end-if
if zc = '1' assign #MITEM-47.enabled := true
end-if
```

```
if zc = '0' assign #MITEM-47.enabled := false
end-if
if zd = '1' assign #MITEM-42.enabled := true
end-if
if zd = '0' assign #MITEM-42.enabled := false
end-if
if x1 = '1'
assign #MITEM-9.enabled := true
assign #MITEM-29.enabled := true
assign #MITEM-35.enabled := false
assign #MITEM-38.enabled := true
assign #MITEM-39.enabled := true
assign #MITEM-40.enabled := true
assign #MITEM-41.enabled := true
assign #MITEM-44.enabled := true
end-if
if x1 = '0'
assign #MITEM-9.enabled := false
assign #MITEM-29.enabled := false
assign #MITEM-35.enabled := false
assign #MITEM-38.enabled := false
assign #MITEM-39.enabled := false
assign #MITEM-40.enabled := false
assign #MITEM-41.enabled := false
assign #MITEM-44.enabled := false
end-if
end-read
```

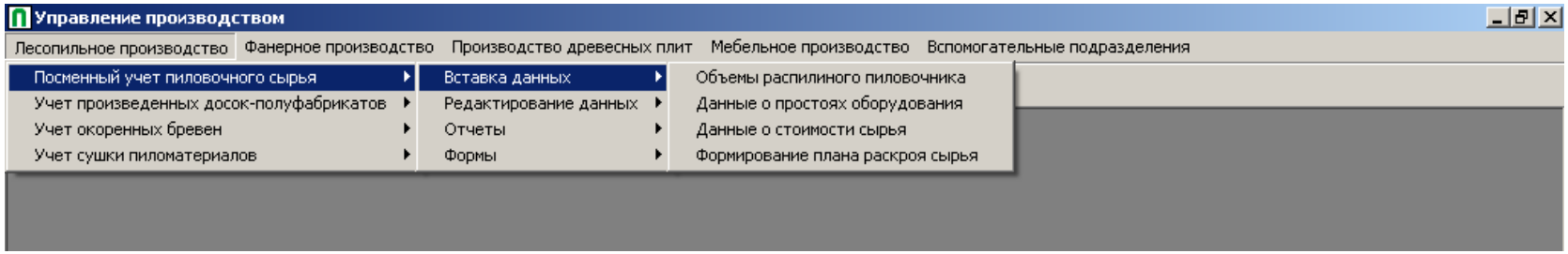


Рис. 3.11. Управляющее меню приложения

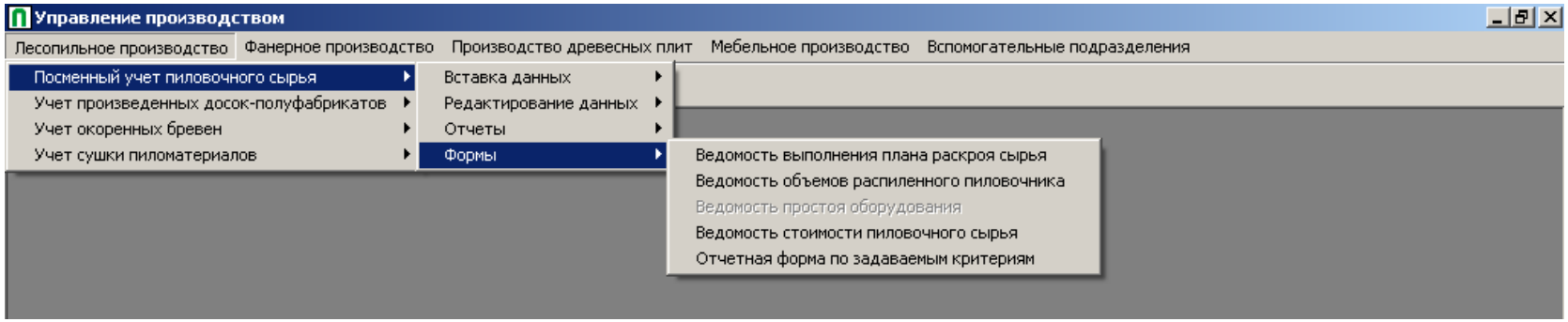


Рис. 3.12. Процедура формирования списка отчетных форм

3.4.3. Автоматическая настройка элементов отчетных форм

Основой для автоматический настройки элементов отчетных форм является электронная форма, которая содержит иерархию сгруппированных индикаторов или переключателей (рис.3.13. и рис. 3.14.). Посредством индикаторов пользователю предлагается активизировать элементы КИС в соответствии с потребностями предприятия и его организационной структурой.

Каждый индикатор (переключатель) может отвечать за один или несколько элементов структуры КИС лесопромышленного предприятия. Индикаторы более низкого уровня активизируются включением индикаторов более высокого уровня. Например, в рамках модуля управления лесопильным производством индикаторы параметров формирования ведомости выполнения плана раскроя пиловочного сырья активизируются посредством индикатора «Ведомость выполнения плана раскроя сырья» - рис. 3.15. и 3.16.

Согласно рис. 3.13. и 3.14. при настройке элементов форм используется включает 6 групп параметров. Первые 5 групп параметров соответствуют видам производств и активизируются в зависимости от тех видов производств, которые существуют на предприятии. Т.о. каждая группа параметров отвечает за настройки специализированных программных блоков КИС (составляющих программных модулей общего назначения). Шестая группа параметров отвечает за дополнительные настройки всех специализированных блоков КИС.

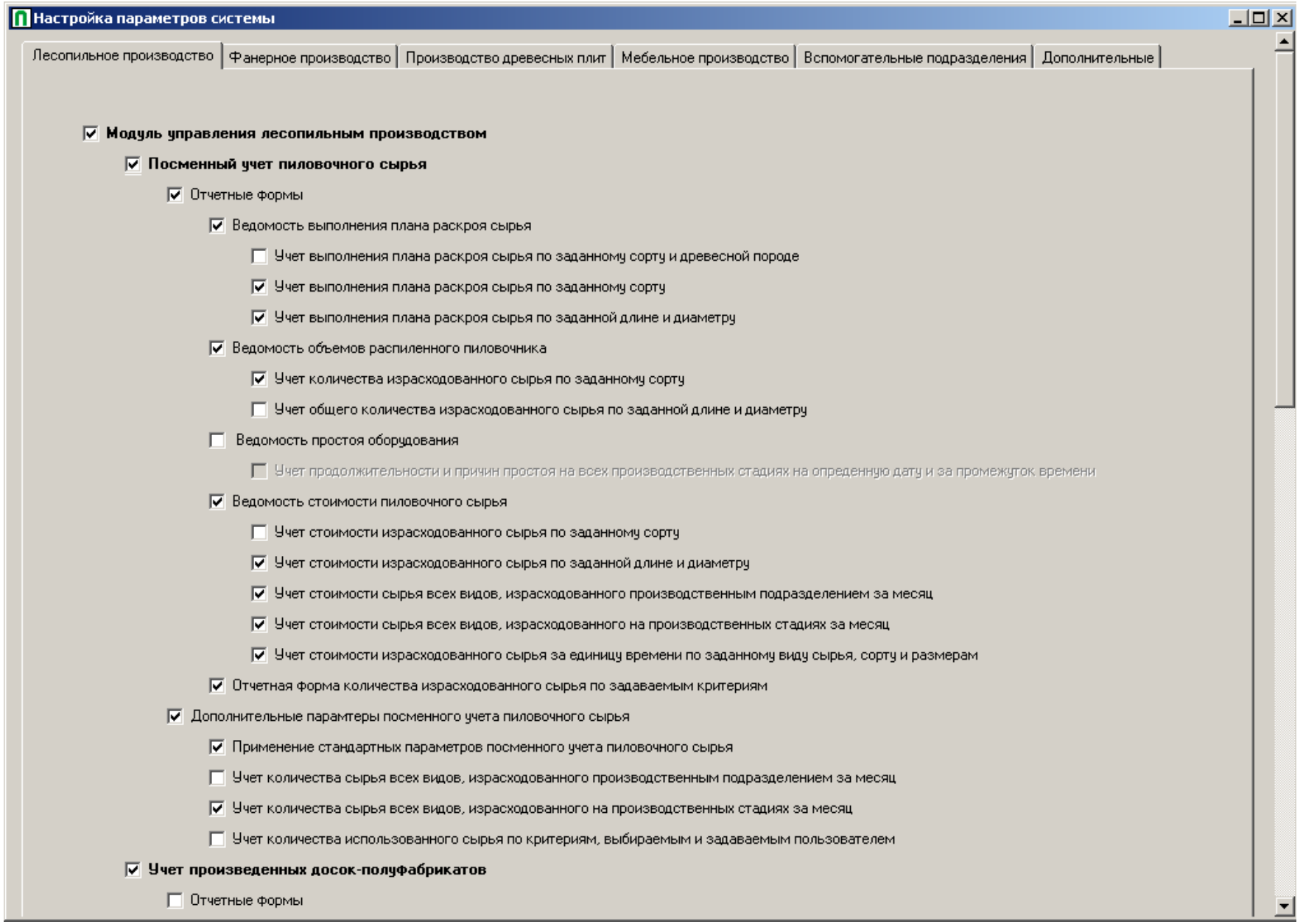


Рис. 3.13. Интерфейс управляющего приложения автоматической настройки элементов отчетных форм

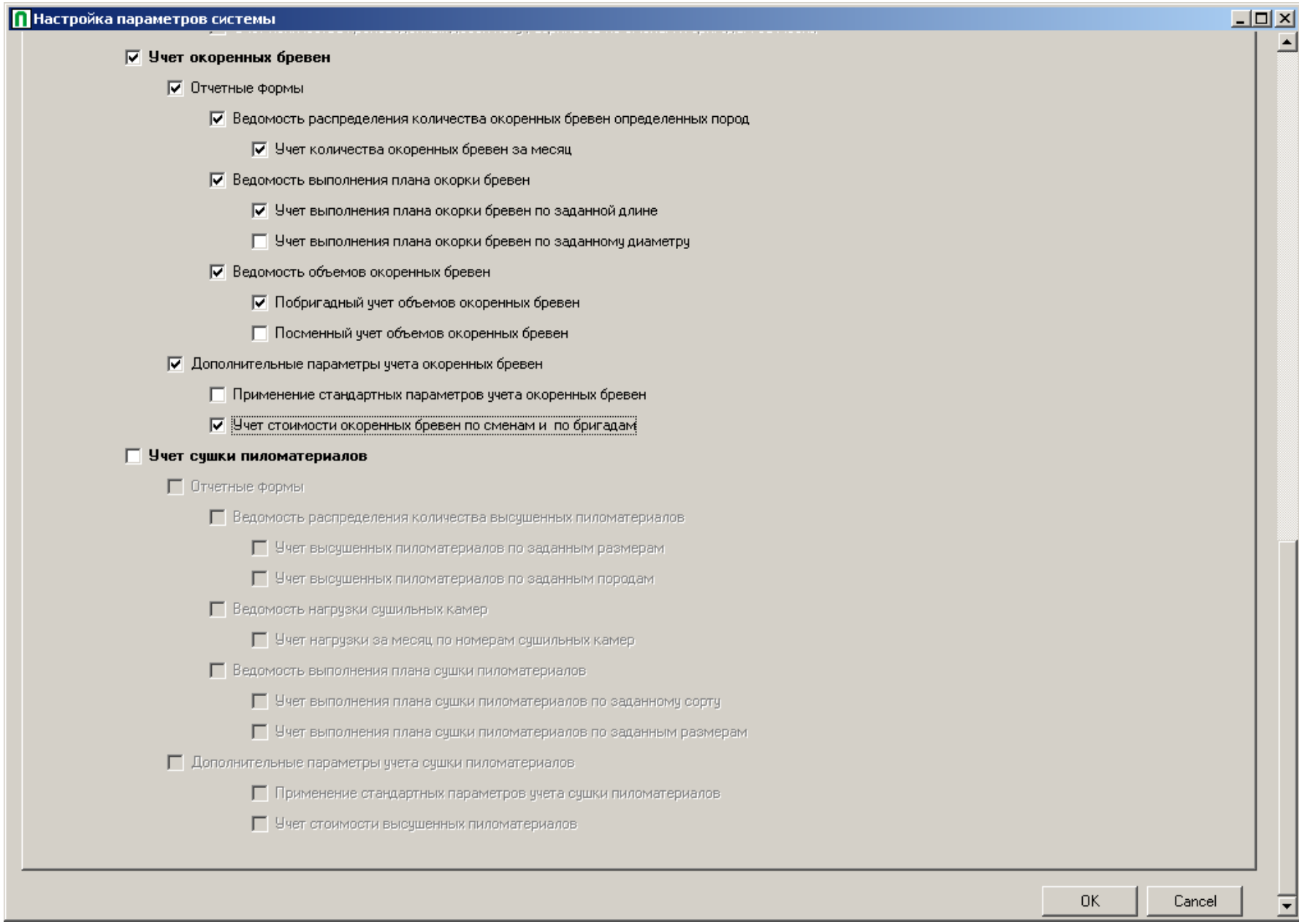


Рис. 3.14. Интерфейс управляющего приложения автоматической настройки элементов отчетных форм

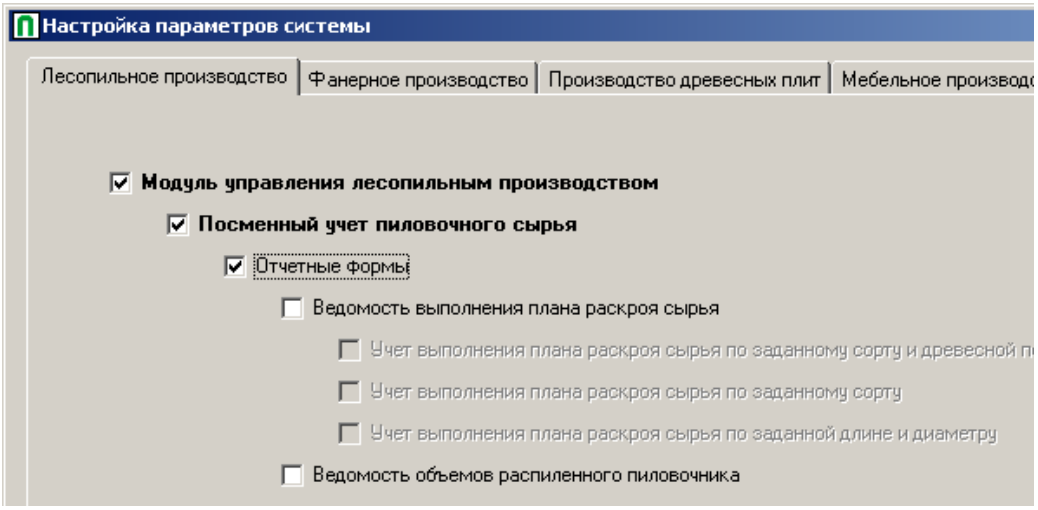


Рис. 3.15. Индикатор «Ведомость выполнения плана раскроя сырья» в состоянии «выключено»

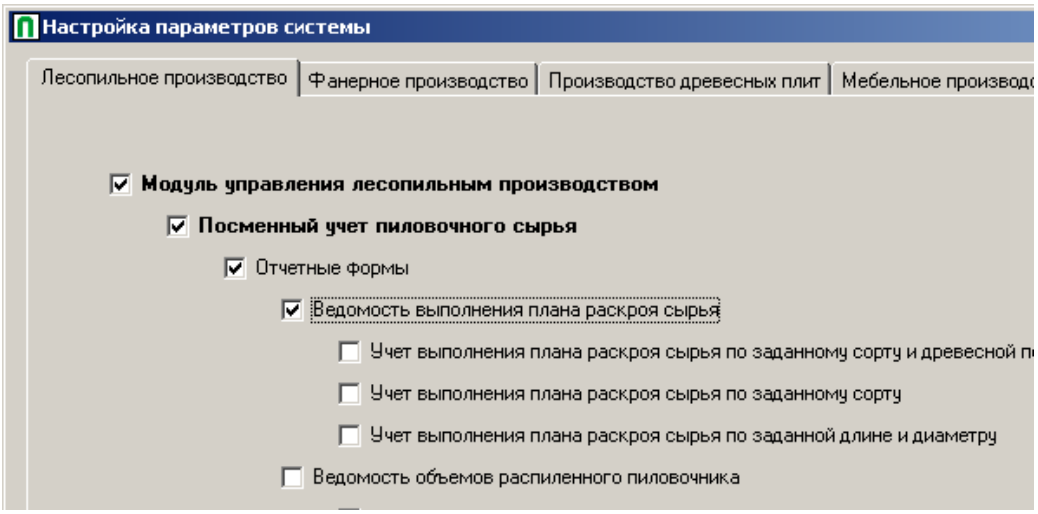


Рис. 3.16. Индикатор «Ведомость выполнения плана раскроя сырья» в состоянии «включено»

Каждая группа индикаторов учета специализированных программных блоков КИС (посменный учет пиловочного сырья и пр.) активизирует индикаторы параметров отчетных форм и индикаторы дополнительных параметров.

3.5. Выводы

1. Существуют 4 основных уровня функционирования элементов КИС лесопромышленного предприятия:

- уровень организации ядра КИС;
- уровень пользовательских приложений КИС;
- уровень расчетных программ КИС;

- уровень отчетных форм.

2. Возможна оптимизация функционирования КИС лесопромышленного предприятия на всех 4-х уровнях, что позволяет снижать затраты на разработку, внедрение и эксплуатацию, а также повышать производительность КИС.
3. Основными критериями оптимальной структуры и состава КИС лесопромышленного предприятия являются:

- эффективная организация ядра (способы задания структуры ядра представлены в п.3.1.);
- оптимизация пользовательских приложений (п.3.2.);
- оптимальное построение запросов и программных кодов (п.3.3.);
- наличие инструмента генерации отчетных форм в соответствии с требованиями конечного пользователя (п.3.4.).

4. Предлагаемые функциональные уровни реализованы в среде, наилучшим образом подходящей для реализации КИС промышленного предприятия (среде СУБД ADABAS и Natural).
5. В предлагаемой среде реализации КИС существует возможность проведения модификаций при помощи графических пользовательских приложений, что обеспечивает простоту и наглядность процесса модификации системы и позволяет снижать затраты на обучение персонала работе с КИС.
6. Одним из наиболее важных компонентов КИС является ядро, которое содержит значения переменных, являющихся основанием для модификации и оптимизации КИС.
7. Основными методами оптимизации КИС лесопромышленного предприятия являются:
   - Метод формирования списка активных элементов КИС;
   - Метод формирования списка активных компонентов КИС;
   - Метод автоматической настройки компонентов КИС;
   - Метод получения отчетных форм на основе задаваемых критериев.

4. АНАЛИЗ ЭФФЕКТИВНОСТИ СИСТЕМНЫХ СВЯЗЕЙ И ЗАКОНОМЕРНОСТЕЙ ФУНКЦИОНИРОВАНИЯ КИС ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

4.1. Основные показатели эффективности различных способов организации ядра КИС лесопромышленного предприятия

В данном разделе приводятся основные показатели эффективности организации ядра КИС способами, описанными в главе 3. База данных о подразделениях и сотрудниках предприятия (рассмотренная в качестве примера в главе 3) организуется в виде каждого из перечисленных способов организации ядра. В качестве основы для создания БД принимаются следующие гипотетические данные:

- количество отделов – 20;
- количество сотрудников – 1000;
- количество заказчиков – 1000;
- количество контрактов – 1000000 (по 1000 у каждого сотрудника).

4.1.1. Реляционная структура ядра

Основные показатели эффективности 1-го способа организации реляционной структуры ядра показана в таблице 4.1.

Основные показатели эффективности реляционной структуры ядра (1-й способ организации)

Таблица 4.1.

Показатель	Кол-во записей в БД	Размер БД, кбайт	Время запроса на выбор-ку, с	Время запроса на подсчет количества записей, с	Время на чтение всех записей в БД, с
1. Кол-во файлов БД - 5	-	-	-	-	-
2. Файл 1 (отделы)	20	4	0,05	0,05	0,1
3. Файл 2 (сотрудники)	1000	64	0,1	0,05	0,4
4. Файл 3 (исполнители)	1000000	16368	0,2	0,05	370,1
5. Файл 4 (заказчики)	1000	108	0,1	0,05	0,4
6. Файл 5 (контракты)	1000000	83762	0,3	0,05	398,6
Общий размер БД	-	100306	-	-	-

Основные показатели эффективности 2-го способа организации реляционной структуры ядра показана в таблице 4.2.

Основные показатели эффективности реляционной структуры ядра (2-й способ организации)

Таблица 4.2.

Показатель	Кол-во записей в БД	Размер БД, кбайт	Время запроса на выборку, с	Время запроса на подсчет количества записей, с	Время на чтение всех записей в БД, с
1. Количество файлов БД (таблиц) - 4	-	-	-	-	-
2. Файл 1 (основное отношение)	1000000	91128	0,3	0,05	412,7
3. Файл 2 (справочная информация об отделах)	20	4	0,05	0,05	0,1
4. Файл 3 (справочная информация о сотрудниках)	1000	60	0,1	0,05	0,3
5. Файл 4 (справочная информация о заказчиках)	1000	108	0,1	0,05	0,4
Общий размер БД	-	91300	-	-	-

4.1.2. Иерархическая структура ядра

Основные показатели эффективности организации иерархической структуры ядра показана в таблице 4.3.

Основные показатели эффективности иерархической структуры ядра

Таблица 4.3.

Показатель	Кол-во записей в БД	Размер БД, кбайт	Время запроса на выборку, с	Время запроса на подсчет количества записей, с	Время на чтение всех записей в БД, с
1. Количество файлов БД (таблиц) - 2	-	-	-	-	-
2. Файл 1 (связь ОТДЕЛЫ-СОТРУДНИКИ-ИСПОЛНИТЕЛИ)	1000000	107268	0,4	0,05	398,2
3. Файл 2 (связь ЗАКАЗЧИКИ-КОНТРАКТЫ-ИСПОЛНИТЕЛИ)	1000000	136216	0,4	0,05	403,3
Общий размер БД	-	243484	-	-	-

4.1.3. Многоуровневая структура ядра

Основные показатели эффективности организации многоуровневой структуры ядра показана в таблице 4.4.

Основные показатели эффективности многоуровневой структуры ядра

Таблица 4.4.

Показатель	Значение
1. Количество файлов БД	1
2. Количество записей в БД	1000000
3. Размер БД, кбайт	82236
4. Время запроса на выборку, с	0,5
5. Время запроса на подсчет количества записей, с	0,05
6. Время на чтение всех записей в БД, с	326,4

4.1.4. Мультипольная структура ядра

Основные показатели эффективности организации мультипольной структуры ядра показана в таблице 4.5.

Основные показатели эффективности мультипольной структуры ядра

Таблица 4.5.

Показатель	Значение
1. Количество файлов БД	1
2. Количество записей в БД	1000000
3. Размер БД, кбайт	86482
4. Время запроса на выборку, с	0,5
5. Время запроса на подсчет количества записей, с	0,05
6. Время на чтение всех записей в БД, с	316,3

4.2. Анализ экономической эффективности внедрения разработанных компонентов и системных связей в КИС лесопромышленного предприятия

Анализ экономической эффективности внедрения модуля базируются на методах расчета совокупной стоимости владения программного обеспечения и корпоративных систем управления [198-201].

Стоимость создания, внедрения и эксплуатирования информационной системы включает следующие компоненты:

1. Общая стоимость всех компонентов программного обеспечения (ПО):

- стоимость лицензий на использование (ПО);
- затраты на обновления ПО;
- затраты на поддержку ПО.

2. Общая стоимость аппаратного обеспечения:

- стоимость ПК пользователей;
- стоимость серверных станций (включая обеспечение электропитанием, серверы тестирования и пр.);

- общая стоимость обновлений и ремонтов аппаратного обеспечения.

3. Общая стоимость администрирование СУБД и аппаратного обеспечения (включая затраты, связанные с внедрением программного продукта).

4. Стоимость обучения персонала организации работе во внедряемой информационной среде.

5. Затраты, связанные с потерей времени персонала организации, при проведении обучения сотрудников.

Таким образом, были использованы следующие формулы.

Совокупная стоимость использования (Total Cost of Ownership):

$$TCO = SWC + HWC + ADC + SLC + TLC \text{, где}$$

SWC – стоимость использования самого программного продукта (Software Cost),

HWC – общая стоимость использования аппаратного обеспечения (Hardware Cost),

ADC – общая стоимость администрирования программного продукта за условленный период (Administrative Cost),

SLC – общая стоимость обучения персонала организации работе с программным продуктом (Staff Learning Cost),

TLC – затраты, связанные с потерей времени, использованного на обучение персонала (Time Losing Cost).

Стоимость использования программного продукта:

$$SWC = \min\left\{\sum_{i=1}^{n} q_i \cdot (p_{ui} + u_{ui} + s_{ui}); \sum_{i=1}^{n} (p_{pi} + u_{pi} + s_{pi}) + \sum_{i=1}^{n} q_i \cdot (p_{ui} + u_{ui} + s_{ui})\right\},$$

где q_i – количество пользователей i-тым приложением программного продукта,

p_{ui} – стоимость лицензии i-того приложения для одного пользователя,

u_{ui} – стоимость обновления i-того приложения для одной лицензии,

s_{ui} – стоимость поддержки i-того приложения для одной лицензии,

p_{pi} – стоимость лицензии i-того приложения для неограниченного количества пользователей,

u_{pi} - стоимость обновления i-того приложения для лицензии неограниченного количества пользователей,

s_{ui} – стоимость поддержки i-того приложения для лицензии неограниченного количества пользователей.

Общая стоимость использования аппаратного обеспечения:

$$HWC = \left(C_{pc} + \sum_{j=1}^{m} C_{paj} \cdot \left(\frac{1}{1+d_j} \right)^{t_j} \right) \cdot q_o + \left(C_s + \sum_{l=1}^{r} C_{sal} \cdot \left(\frac{1}{1+d_l} \right)^{t_l} \right)$$

, где

C_{pc} – стоимость приобретения одного персонального компьютера для пользователя,

C_{paj} – стоимость последующего j-го обновления или ремонта компьютера,

d_j , d_l – учетные ставки (ставки рефинансирования) за периоды tj и tl,

q_o – общее количество пользователей продуктом,

C_s – стоимость приобретения серверного оборудования (зависит от количества пользователей и желаемого качества работы программного продукта),

C_{sal} - стоимость последующего l-го обновления или ремонта серверного оборудования.

При расчете стоимости обновлений и ремонта, опираясь на данные статистики, считаем, что за 2 года использования восстановительная стоимость компьютера составит около 70% его первоначальной стоимости. Обновления компьютерного парка в данном случае планируются с частотой 1 раз в квартал, т.е. всего 8 обновлений (m=8, r=8). Ставки рефинансирования за последующие периоды были спрогнозированы методом экспоненциального сглаживания.

Общая стоимость внедрения и администрирования программного продукта:

$$ADC = g \cdot \sum_{k=1}^{z} A_{qk} \cdot \left(\frac{1}{1+d_k} \right)^{t_K}$$, где

A_{qk} – стоимость администрирования данного программного обеспечения одним сетевым администратором при количестве пользователей q за 1 месяц в текущих ценах,

d_k – учетная ставка за период t_k,

g – необходимое количество администраторов для функционирования системы.

В данном расчете в качестве базовых были использованы средние данные рынка труда о заработной плате администраторов программного обеспечения. Последующие значения оплаты администрирования за месяц были скорректированы с учетом среднегодового роста средней заработной платы в РФ.

Стоимость обучения персонала организации работе с программным обеспечением:

$$SLC = a \cdot q_o \cdot b \cdot e \cdot \left(\frac{1}{1+d_0} \right)^{t}$$, где

a – количество часов в неделю, затрачиваемых на обучение одним пользователем,

b – количество недель, необходимых для обучения пользователей,

e – средняя стоимость одного часа обучения,

d_o – учетная ставка на период обучения.

Считаем, что период обучения составит 4 часа в неделю на протяжении 4 недель через пол года после приобретения и внедрения программного продукта.

Стоимость времени затраченного персоналом на:

$$TLC = \sum_{h=1}^{y} a \cdot b \cdot x_h \cdot \left(\frac{1}{1+d_o} \right)^{t}$$, где

x_h – оплата 1 часа повседневной работы h-го пользователя.

Затраты на внедрение и администрирование информационной системы являются одними из наиболее весомых. На рис. 4.1. изображена диаграмма сравнения средних стоимостей использования СУБД и редакторов приложений 3-х ведущих фирм-поставщиков на 1 человека по статьям затрат. Все показанные далее стоимостные величины измеряются в условно принятых единицах.

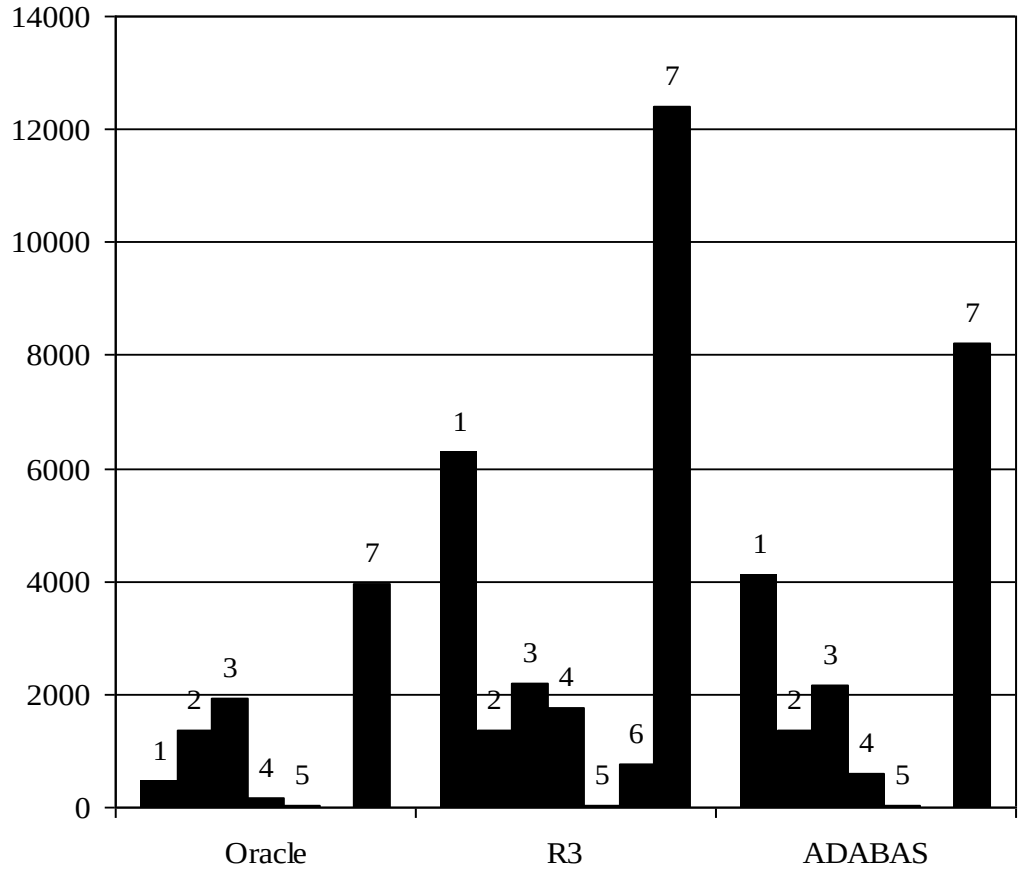


Рис. 4.1. Совокупная стоимость использования СУБД и редакторов приложений (по статьям затрат)

На рис. 4.1. введены следующие условные обозначения:

1 - стоимость использования программного обеспечения

2 – стоимость использования аппаратного обеспечения

3 – стоимость администрирования

4 – стоимость обучения персонала

5 – затраты, связанные с обучением работников

6 – консультационные услуги

7 – совокупная стоимость использования СУБД и редакторов приложений

Согласно вышеуказанных формул была подсчитана совокупная стоимость использования СУБД ADABAS и редактора приложений Natural при построении КИС лесопромышленного предприятия (таблица 4.6.).

Как видно из таблицы 4.7., затраты на внедрение КИС лесопромышленным предприятием напрямую зависят от количества пользователей системы. На рис. 4.2. показан график зависимости величины затрат на внедрение КИС лесопромышленного предприятия в среде ADABAS и Natural от численности персонала (количества пользователей КИС).

Величина затрат на внедрение КИС лесопромышленного предприятия составляет в среднем 36% от общей совокупной стоимости использования СУБД ADABAS и редактора Natural (таблица 4.7.).

Таким образом, использование модуля модификации позволяет сократить расходы на внедрение КИС лесопромышленного предприятия на 32% и более.

Совокупная стоимость использования СУБД ADABAS и редактора Natural

Таблица 4.1.

Затраты	Количество пользователей внедряемой системы										
	2	5	10	25	50	100	200	300	500	800	1000
1	2	3	4	5	6	7	8	9	10	11	12
1. Стоимость ПО СУБД ADABAS и редактора Natural	8267	20667	36333	78333	136667	293333	526667	840000	1266667	2106667	2633333
2. Стоимость аппаратного обеспечения	3988	7588	13587	34445	64441	127611	260306	385058	648860	1057749	1297719
3. Внедрение и администрирование	30607	30607	40316	67651	125571	231142	470436	720782	1249095	1777408	2141565
4. Обучение персонала	1169	2923	5847	14617	29233	58466	116932	175399	292331	467730	584662
5. Потери времени	94	234	468	1169	2339	4677	9355	14032	23386	37418	46773
Итого:	**44125**	**62019**	**96551**	**196215**	**358251**	**715229**	**1383696**	**2135271**	**3480339**	**5446972**	**6704052**

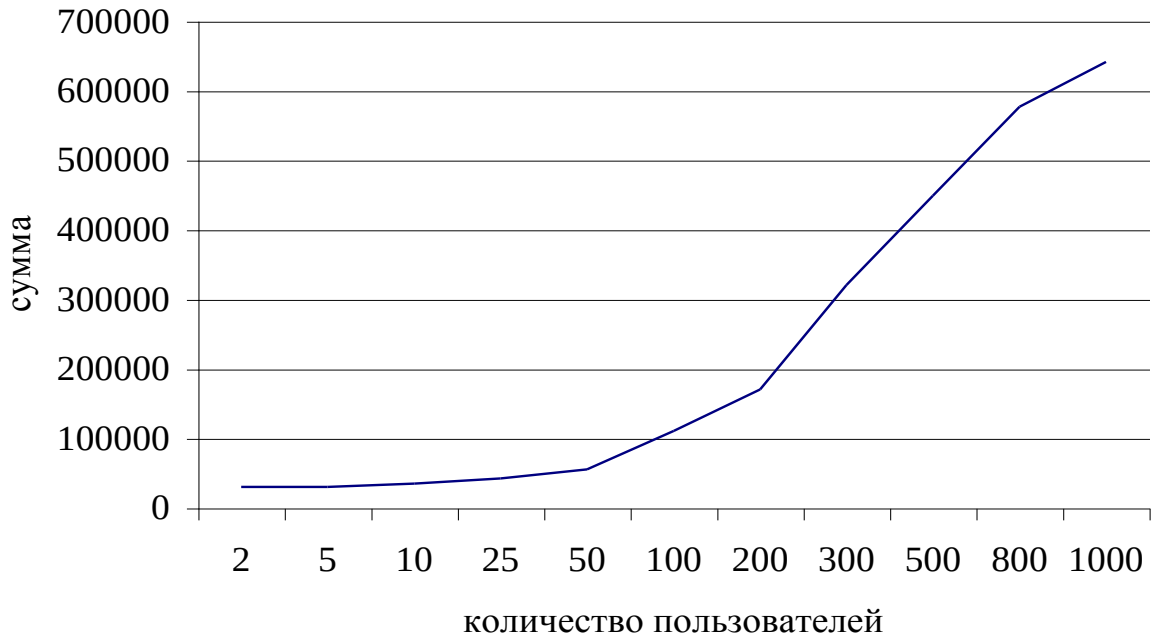


Рис. 4.11. Величина затрат на внедрение КИС лесопромышленного предприятия

4.3. Выводы

1. Предложенный в монографии метод модификации КИС лесопромышленного предприятия реализован в наиболее подходящей среде в виде программного модуля, с использованием наиболее эффективных и наиболее точно отражающих особенности лесопромышленного предприятия структур хранения данных, методов доступа к данным и архитектуры системы.

2. Согласно проведенного анализа эффективности использования предложенного метода позволяет сократить затраты на внедрение и эксплуатацию КИС на 32% и более.

3. Реализованный метод модификации позволяет оптимизировать модульные структуры КИС, а также адаптировать их в соответствии с изменяющейся структурой организации, производственными и технологическими изменениями.

4. Эффективность предложенного метода подтверждается апробацией результата путем моделированием процесса модификации с различными характеристиками, а также практическим опытом реализации модуля модификации КИС лесопромышленного предприятия.

Доля затрат на внедрение КИС лесопромышленного предприятия

Таблица 4.2.

Количество пользователей КИС	2	5	10	25	50	100	200	300	500	800	1000
Внедрение и администрирование	30607	30607	40316	67651	125571	231142	470436	720782	1249095	1777408	2141565
Совокупная стоимость использования ПО	44125	62019	96551	196215	358251	715229	1383696	2135271	3480339	5446972	6704052
Доля затрат на внедрение и администрирование, %	69,36	49,35	41,76	34,48	35,05	32,32	34,00	33,76	35,89	32,63	31,94

5. ОПРЕДЕЛЕНИЕ ЛОГИЧЕСКОЙ И ФИЗИЧЕСКОЙ СТРУКТУРЫ БД ADABAS

5.1. Определение БД

В рамках создания информационной системы средствами ADABAS и Natural первым необходимым шагом является определение структуры базы данных (хранилища данных).

Процедура определения базы данных включает три операции: физическое размещение, создание и редактирование файлов-контейнеров базы данных (файлы с названиями ASSO, DATA, WORK).

Для создания новой базы данных следует в главном контекстном окне (DBA Workbench) выбрать меню Database → New (рис. 5.1.)

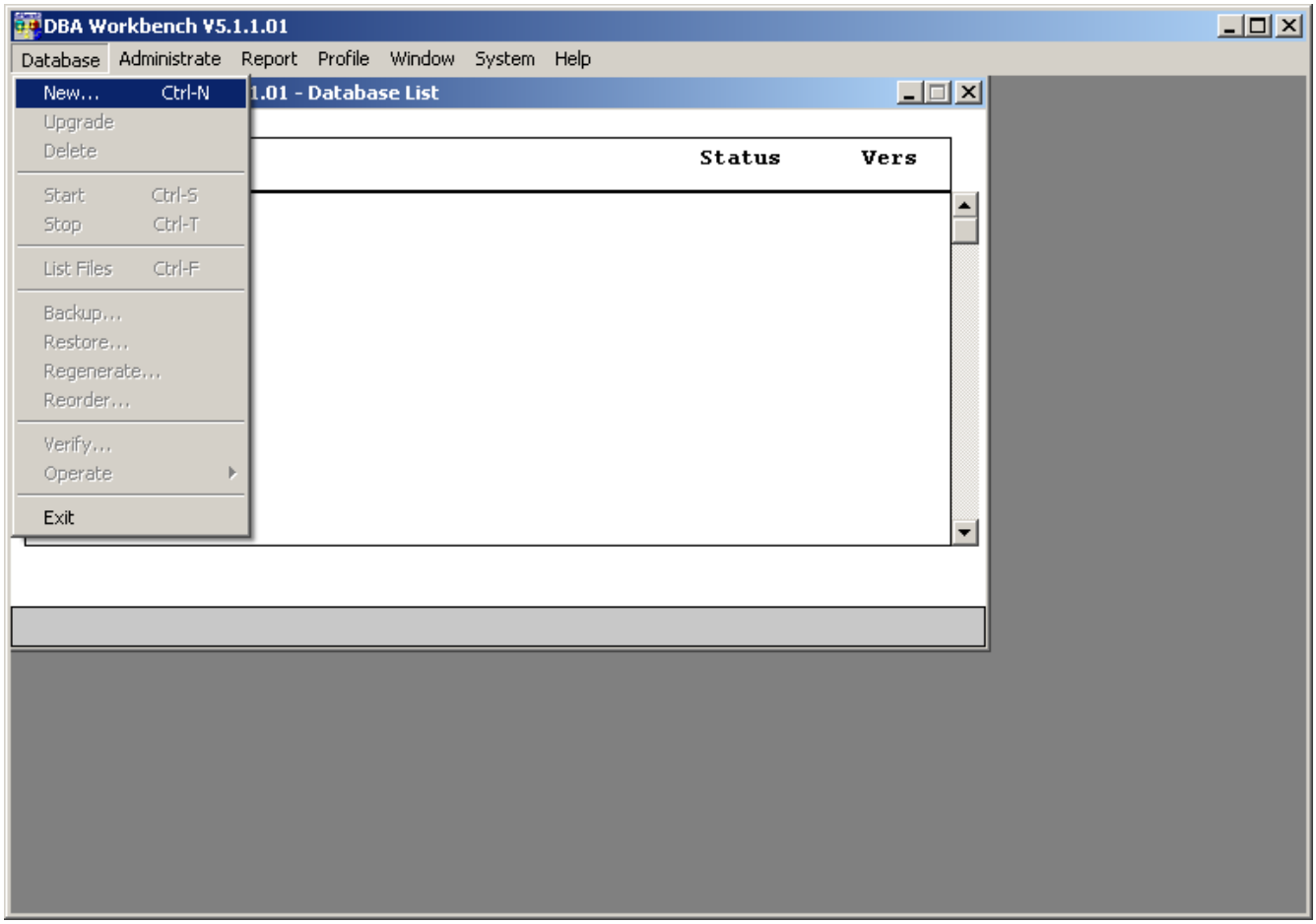


Рис 5.1. Создание новой БД

В последующем окне (‘Create Database’ - рис 5.2.) для изменения будет предложен ряд параметров, определяющих размер и свойства файлов-

контейнеров и БД. Все значения параметров и БД могут быть сохранены по умолчанию.

Все параметры, предложенные для изменения в появившемся окне, сгруппированы в 3 группы: Информация о базе данных (Database Information), Параметры файлов структуры (ASSO, DATA, WORK), Прочие параметры (Miscellaneous).

Ниже подробно рассмотрены параметры каждой из указанных групп.

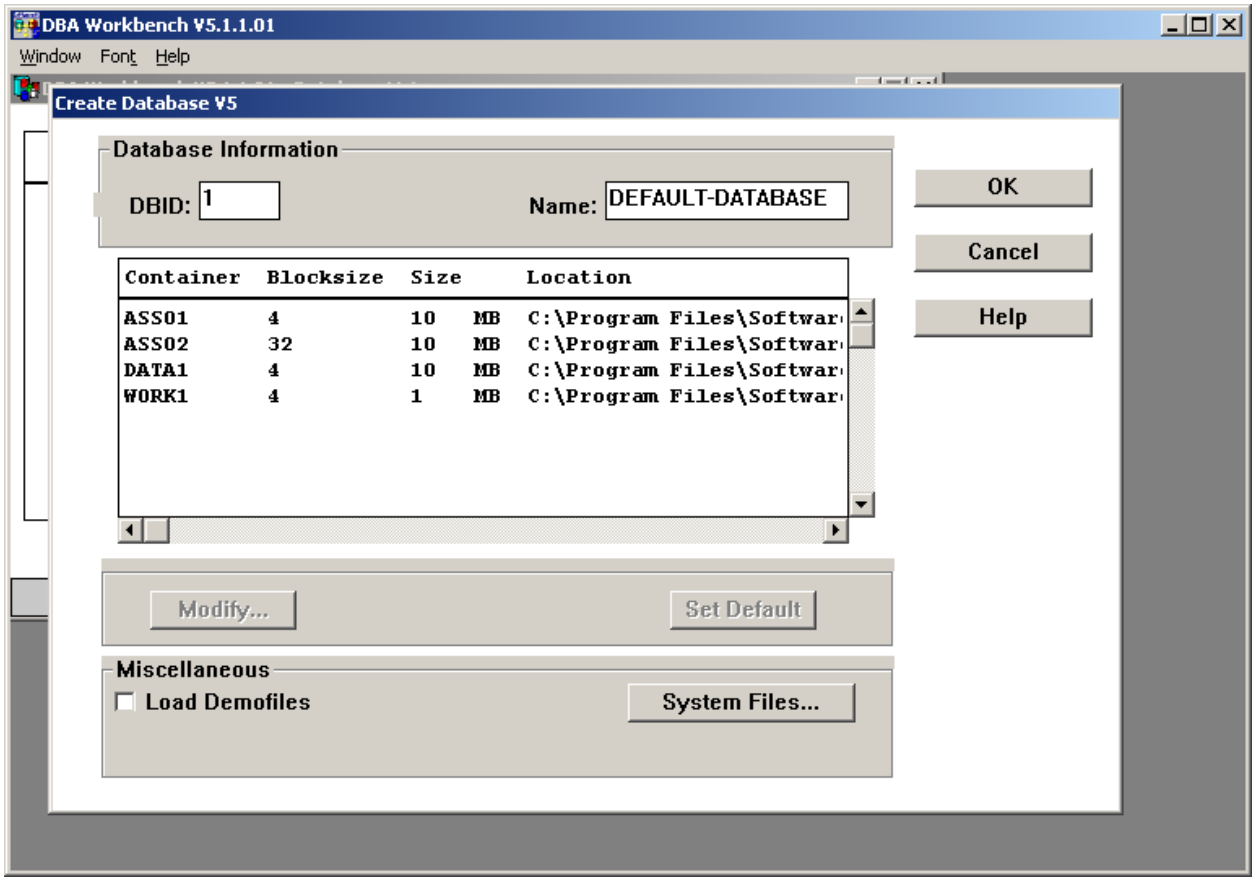


Рис 5.2. Параметры БД

Информация о базе данных (Database Information)

Идентификационный номер БД (DBID)

При открытии окна (рис 5.2.) курсор мыши автоматически устанавливается на поле со списком ‘DBID’ (идентификационный номер БД). Следует оставить неизменным номер, предложенный системой, или ввести иной номер при помощи клавиатуры. Также можно воспользоваться списком данного поля и выбрать номер из списка. Следует учитывать, что системой в

качестве идентификационного номера БД (‘DBID’) принимается любой номер от 1 до 255, который еще не был использован для создания другой БД.

Название БД (Name)

При определении БД в поле ‘Name’ необходимо ввести название БД. Название не должно превышать 16 символов. Также можно сохранить название, предлагаемое по умолчанию.

Параметры файлов структуры (ASSO, DATA, WORK)

Изменение параметров файлов структуры, предложенных системой, следует производить посредством выбора соответствующего файла (ASSO1, ASSO2, DATA1, WORK1 и пр.) при помощи левой кнопки мыши и последующего выбора опции ‘Редактировать’ (‘Modify…’) – рис. 5.2. При этом системой будет предложено редактирование следующих параметров файлов (рис. 5.3.).

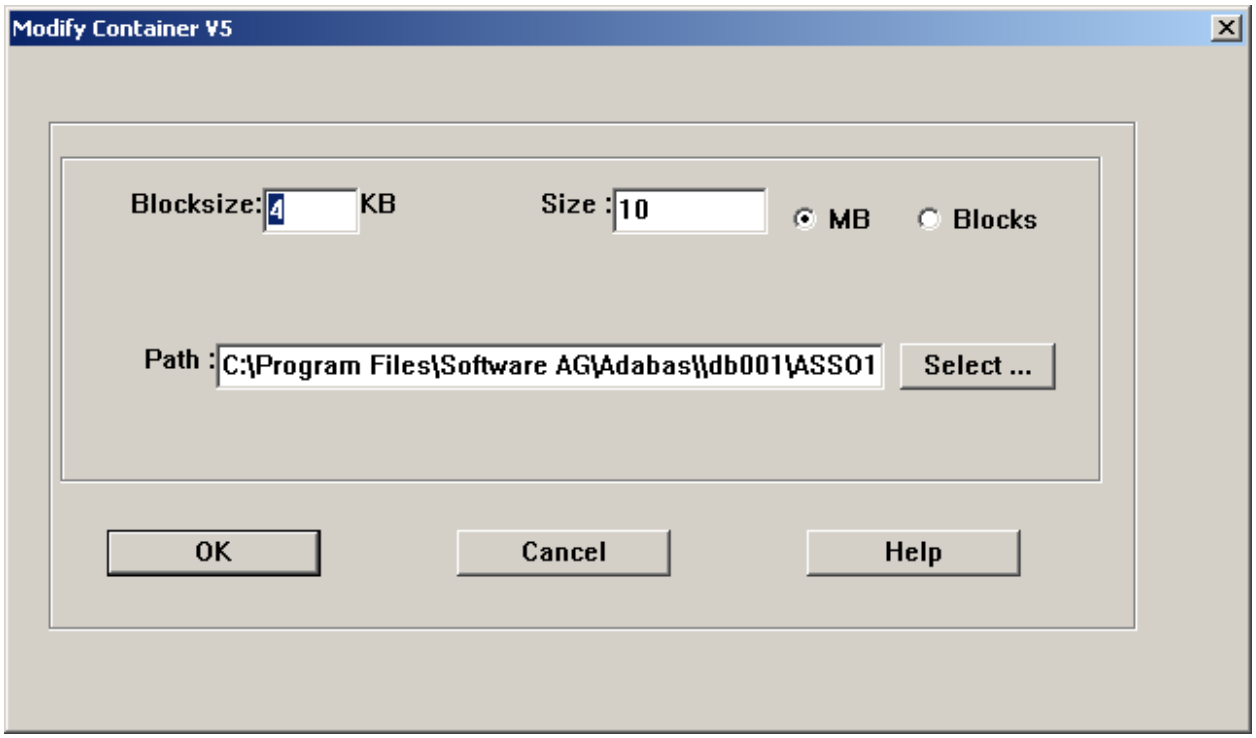


Рис. 5.3. Изменение параметров файлов структуры

Размер блока файла (Blocksize)

В данном поле (‘Blocksize’) вводится размер блока для файла-контейнера ассоциатора БД (единица измерения – килобайт). Если значение

данного поля не определено, то системой будет использоваться значение, предложенное по умолчанию.

Замечание! При определении значения данного параметра для файла ассоциатора ('ASSO Blocksize') необходимо учитывать, что размер блока файла ассоциатора (ASSO) должен быть меньше размера блока рабочего файла (WORK). В противном случае значение данного поля не будет приняты системой, о чем проектировщик БД будет информирован следующим сообщением (рис 5.4.):

Рис 5.4. Информационное сообщение, получаемое при превышении размера файла ассоциатора над размером рабочего файла

Размер файла (Size)

Посредством ввода значения в поле 'Size' определяется размер дискового пространства, используемого для изменяемого файла. Размер дискового пространства можно определить в мегабайтах (опция 'MB') и в блоках (опция 'Blocks'). Значение данного параметра, предлагаемое по умолчанию – 1 Мб. Для увеличения дискового пространства для файла ассоциатора необходимо ввести соответствующее значение.

Расположение файла (Path)

В текстовом поле 'Path' отображается полная спецификация локализации файла ассоциатора. Можно оставить спецификацию неизменной (как предложено системой по умолчанию), либо установить иную область локализации файла при помощи вызова обзорного окна (кнопка 'Select', как показано на рис 5.3.) – рис 5.5.

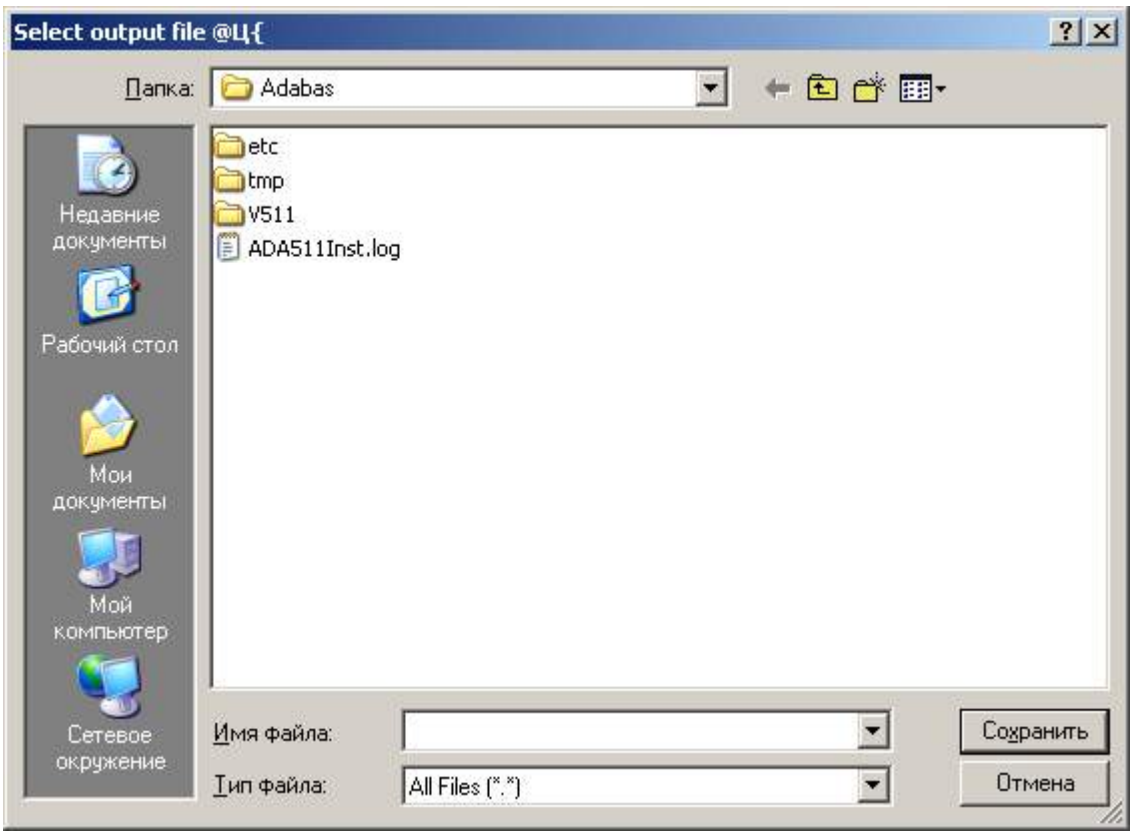


Рис 5.5. Определение локализации файла

Для установки параметров файлов структуры на уровне значений, предлагаемых системой по умолчанию следует воспользоваться опцией ‘Установить по умолчанию’ (‘Set Default’) – рис. 5.2.

Прочие параметры (Miscellaneous)

Номера системных файлов (System Files)

В рамках данного подраздела должны быть определены идентификационные номера системных файлов БД (файлы Checkpoint File, Security File, User Data File), которые автоматически создаются и загружаются при создании БД. При нажатии на командную кнопку ‘Miscellaneous System Files…’ появляется следующее окно (рис. 5.6.):

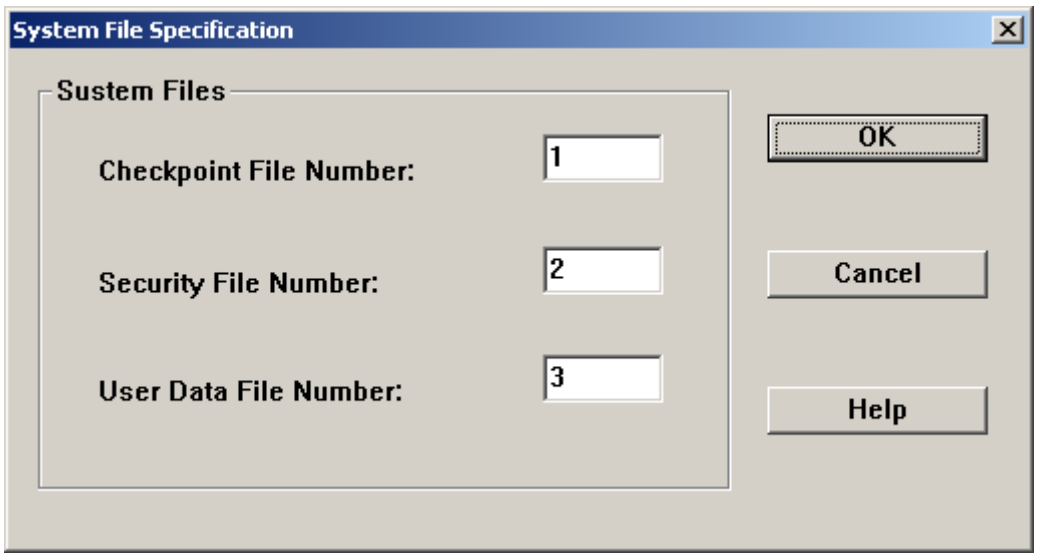


Рис. 5.6. Прочие параметры

По умолчанию (как показано на рис 5.6.) системой предлагаются следующие номера системных файлов: checkpoint – 1, security – 2, user data – 3). При необходимости можно изменить номера системных файлов. Для сохранения изменений нажмите OK.

Замечание! Номера системных файлов не должны превышать максимальное число файлов БД (‘Miscellaneous Maximum No. of Files’). В противном случае произведенные изменения не будут сохранены системой, о чем проектировщик БД будет проинформирован следующим сообщением (рис. 5.7.):

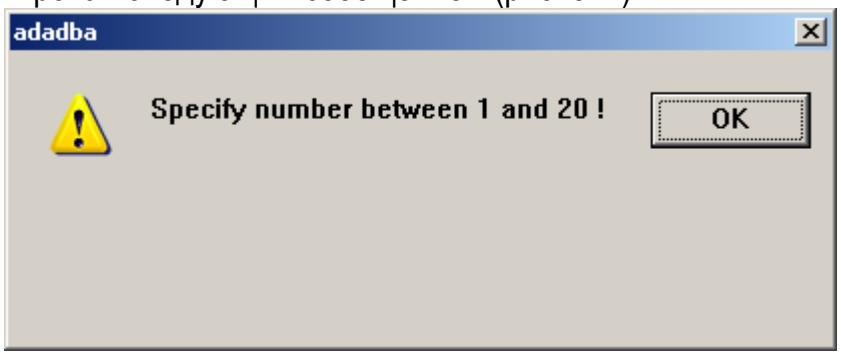


Рис. 5.7. Информационное сообщение, получаемое при превышении максимального числа файлов БД

Загрузка демо-файлов (Load Demofiles)

Для загрузки демонстративных файлов, предлагаемых системой, следует поставить флажок на опции ‘Загрузка демо-файлов’ (‘Load Demofiles’) – рис. 5.2. При этом после завершения процедуры создания БД в структуру БД будут загружены соответствующие файлы.

После ввода и редактирования указанных параметров (Информация о базе данных (Database Information), Параметры файлов структуры (ASSO, DATA, WORK), Прочие параметры (Miscellaneous)) для определения новой БД следует в окне ‘Create Database’ выбрать ‘OK’.

Если вышеописанные параметры были заданы корректным образом, проектировщик будет проинформирован о создании новой БД следующим сообщением (рис. 5.8.):

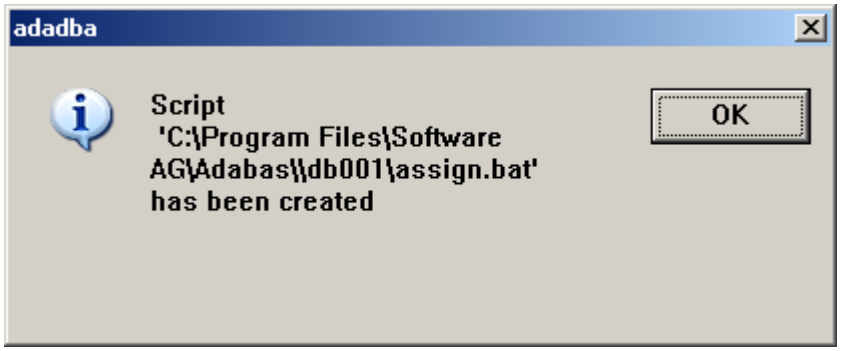


Рис. 5.8. Информационное сообщение об успешном создании БД

Вид главного управляющего окна (‘DBA Workbench’) поменяется следующим образом (рис. 5.9.):

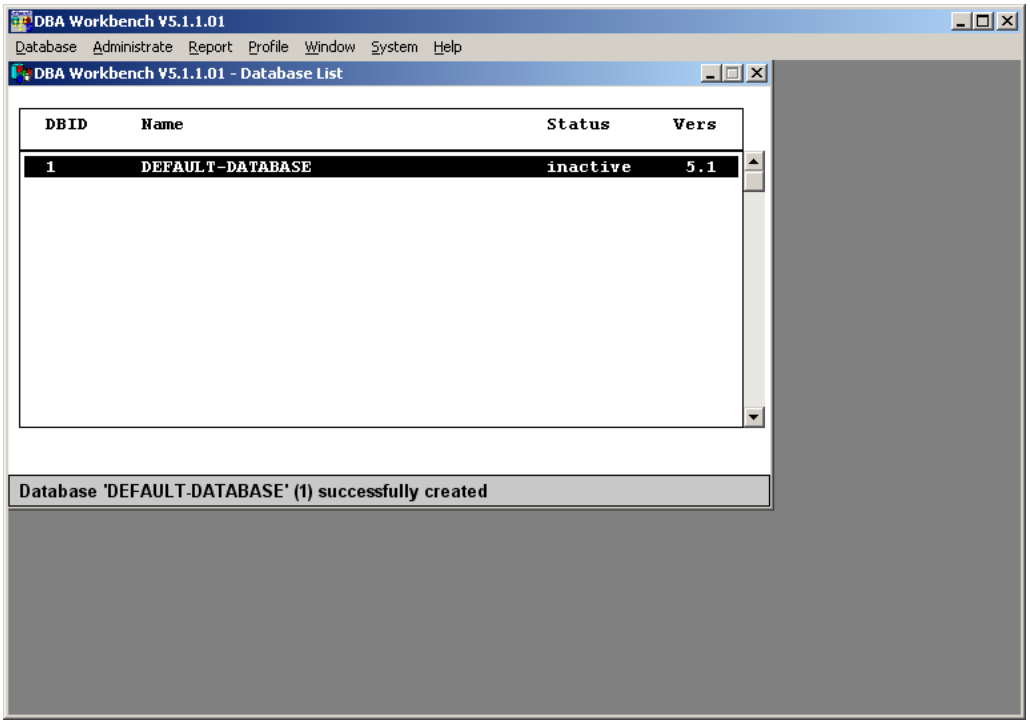


Рис. 5.9. Вид главного управляющего окна после создания новой БД

5.2. Определение файла БД

После определения БД для обеспечения возможности осуществления трансакций (операции внесения, удаления, редактирования данных) необходимо определение структуры файла в рамках созданной БД.

В системе ADABAS определение файла подразумевает создание ***таблицы определения данных (Field Definition Table – FDT)*** с последующим определением свойств файла (локализация дискового пространства файла и пр.). После создания FDT и определения параметров файл создается в рамках выбранной БД и отображается в списке файлов БД.

Таблица FDT создается при помощи редактора FDT, а также может быть прочитана из уже существующего файла FDT, или файла ***модуля определения данных (Data Definition Module – DDM)*** конструктора Natural.

Для создания нового файла в рамках существующей БД необходимо в главном управляющем окне (‘DBA Workbench’) выбрать (щелчком мыши) БД, в список файлов которой необходимо включить определяемый файл (рис. 5.10).

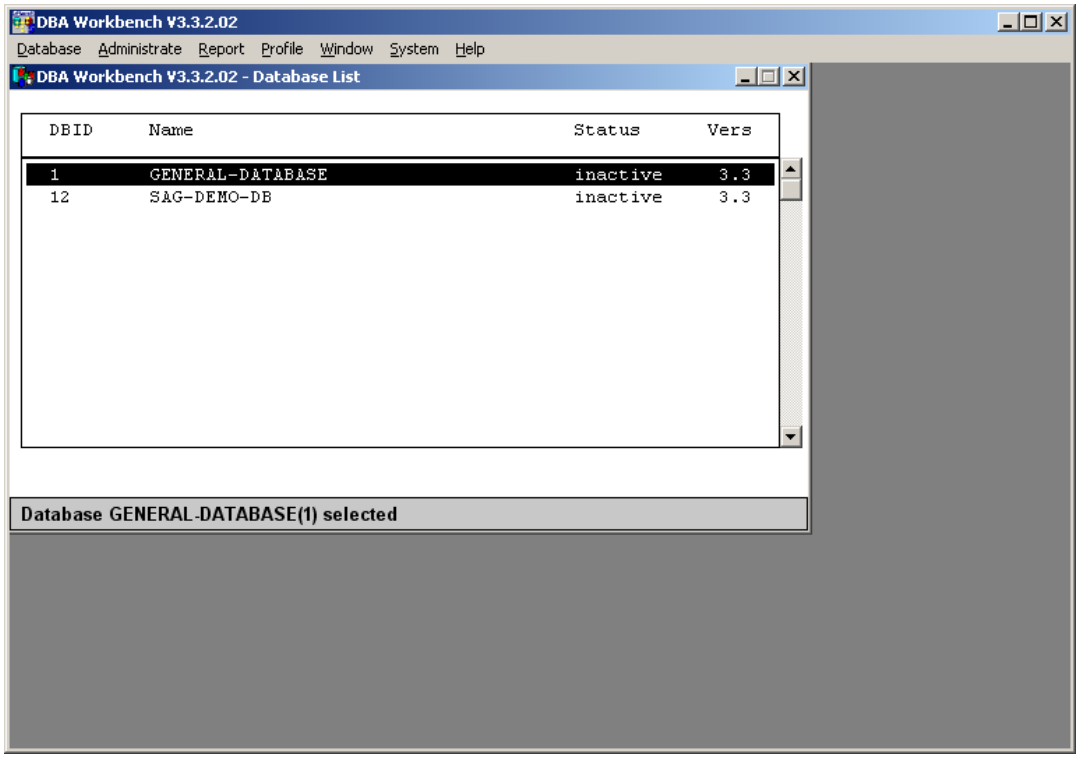


Рис. 5.10. Выбор БД для создания файла БД

Далее необходимо открыть окно списка существующих файлов данной БД. Окно списка открывается либо двойным щелчком мыши по записи выбранной БД, либо посредством меню Database → List Files (рис. 5.11).

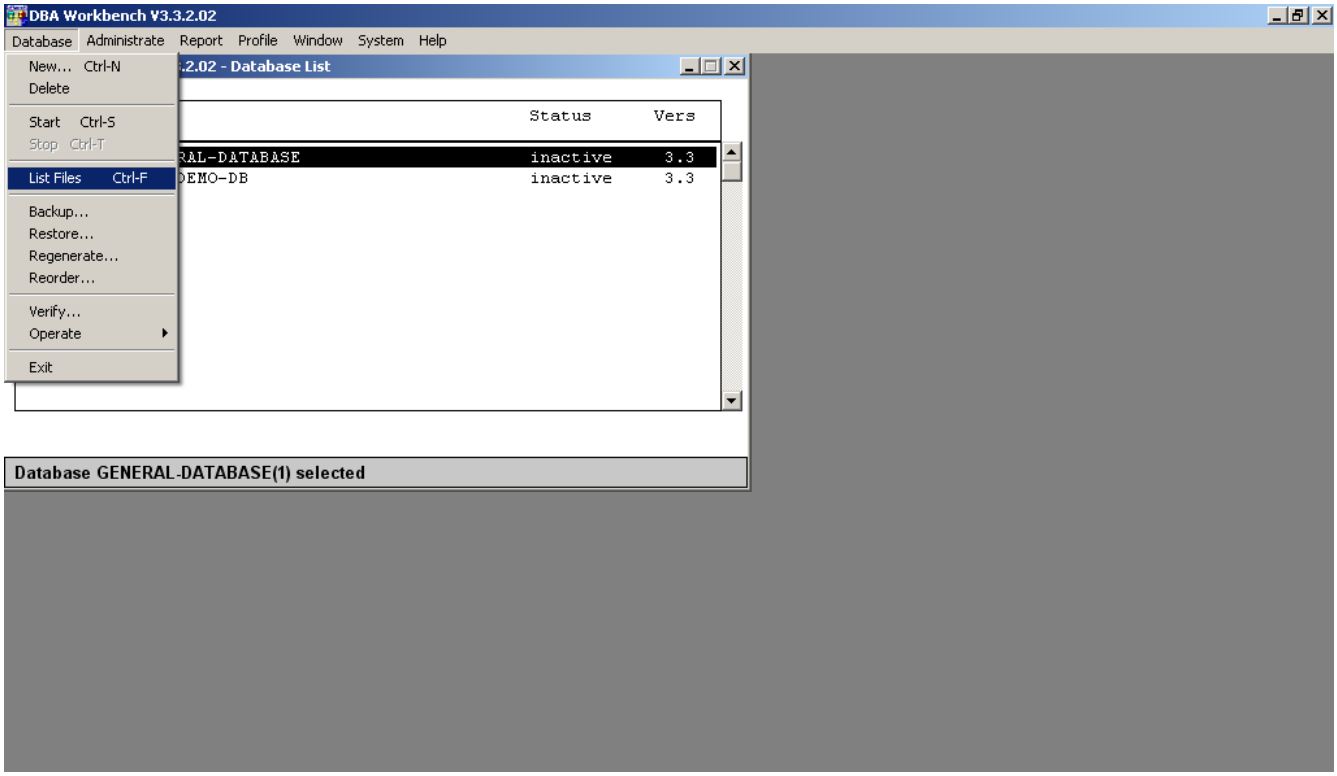


Рис. 5.11. Просмотр списка уже существующих файлов в БД

В появившемся окне списка файлов, даже если ни одного файла не было создано, будут отображаться 3 системных файла (рис. 5.12.). Данные файлы создаются автоматически при определении БД.

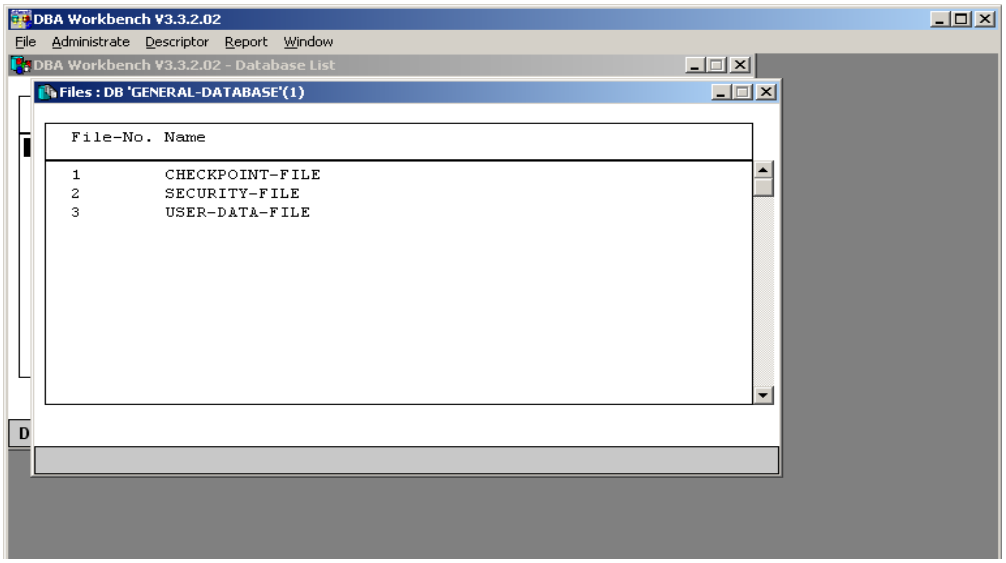


Рис. 5.12. Список уже существующих файлов БД

Как показано на рис. 5.12., системные файлы имеют первые три номера файла. Эти номера присваиваются по умолчанию при определении БД, и могут быть изменены на другие в посредством ‘Miscellaneous System Files…’.

Далее следует выбрать в контекстном меню File → New (рис. 5.13.).

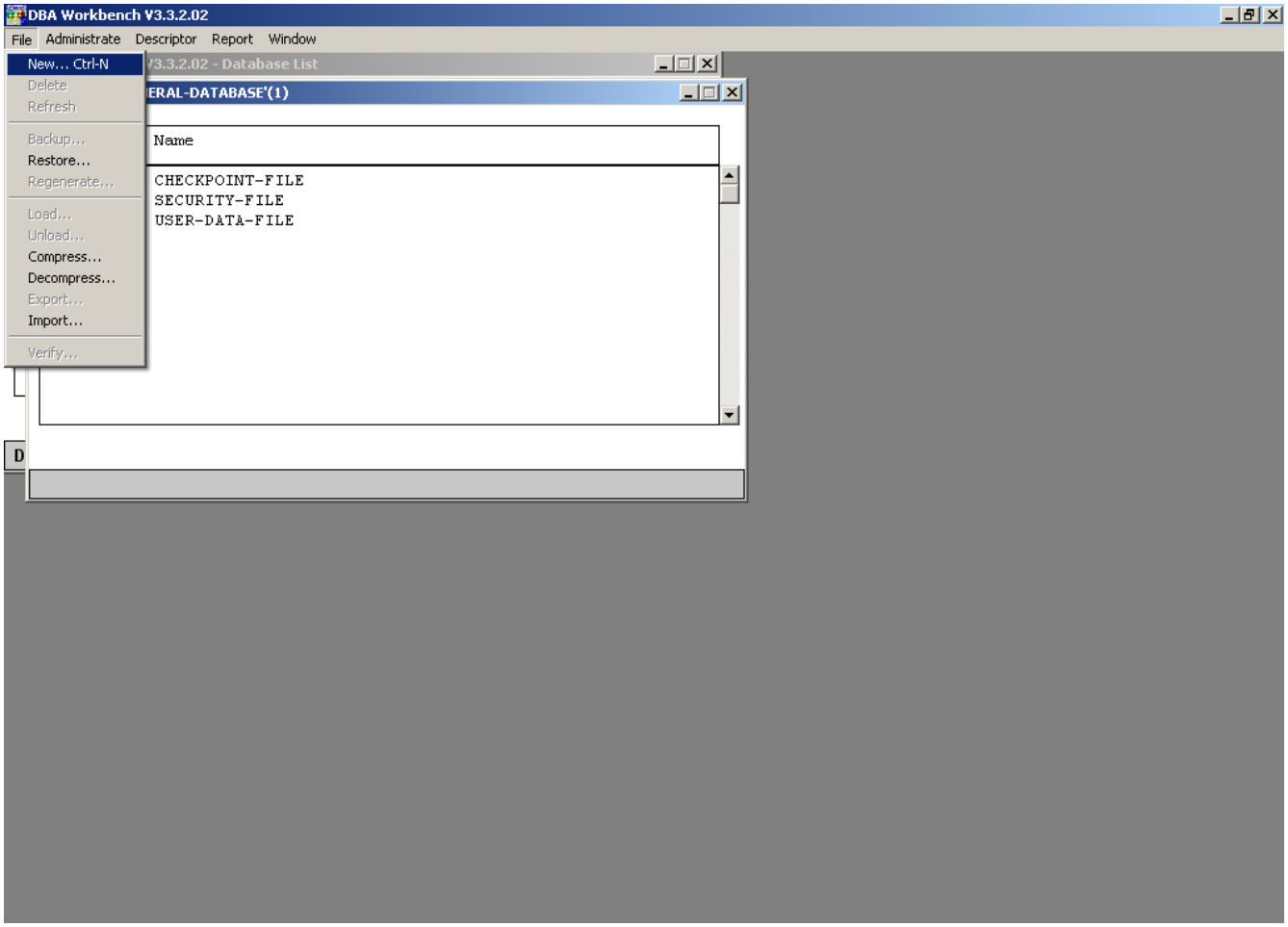


Рис. 5.13. Создание нового файла БД

После проведения данной процедуры открывается окно редактора FDT (рис. 5.14). При помощи данного редактора осуществляется определение составляющих файла (записи и элементов) посредством ввода следующей информации в текстовые поля и поля со списком: тип элемента, уровень элемента, сокращенное имя элемента, длину элемента (в байтах), формат данных элемента и опции ограничения элемента записи. Кроме того, любой из элементов, определяемый в рамках файла может быть объявлен дескриптором (простым дескриптором, фонетическим дескриптором, субдескриптором, супердескриптором или гипердескриптором).

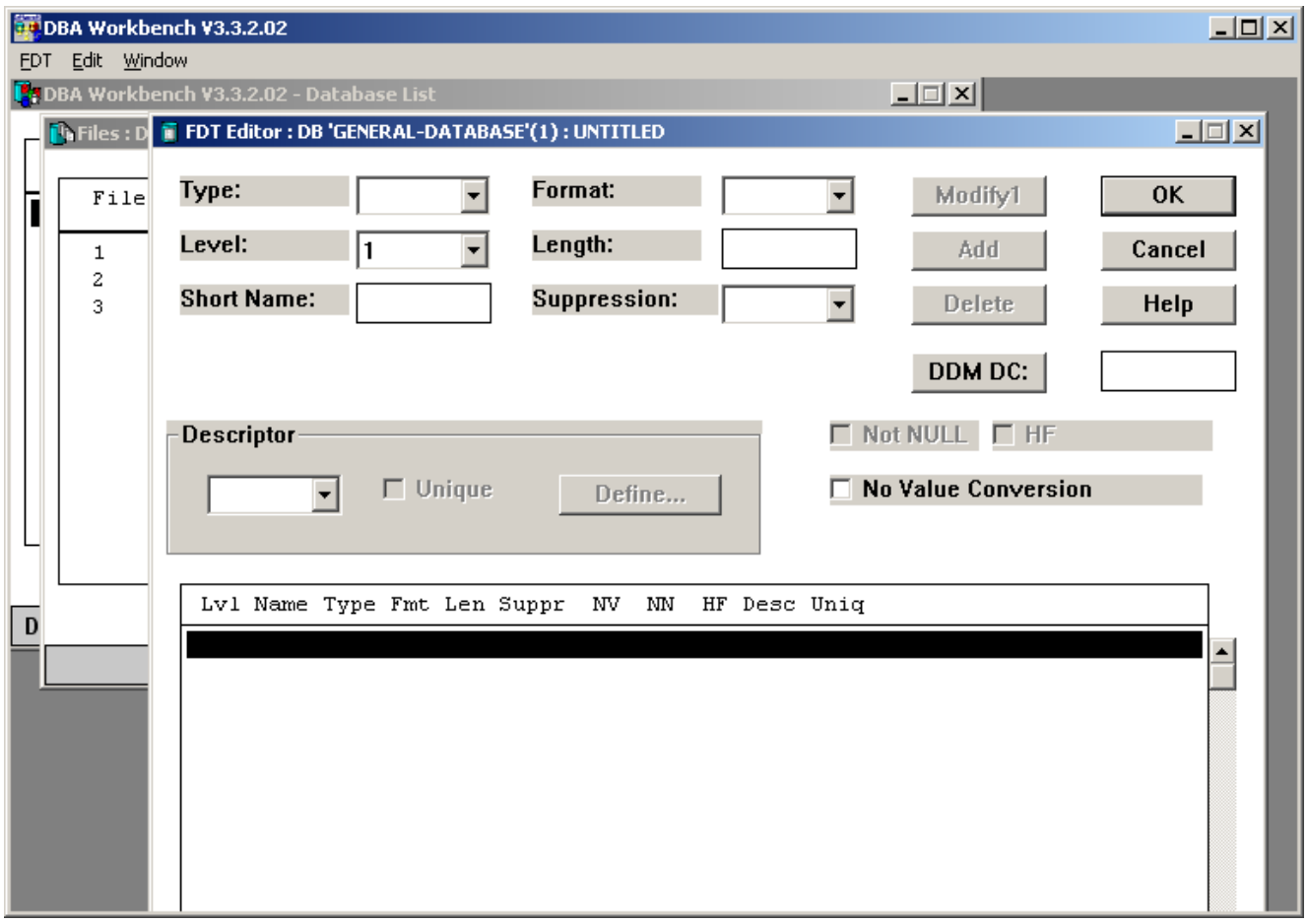


Рис. 5.14. Окно редактора БД

После определения составляющих файла для сохранения всех введенных значений элементов в структуре файла необходимо нажать ‘OK’. При этом открывается окно ввода параметров, необходимых для создания файла (‘Create File’) – рис. 5.15.

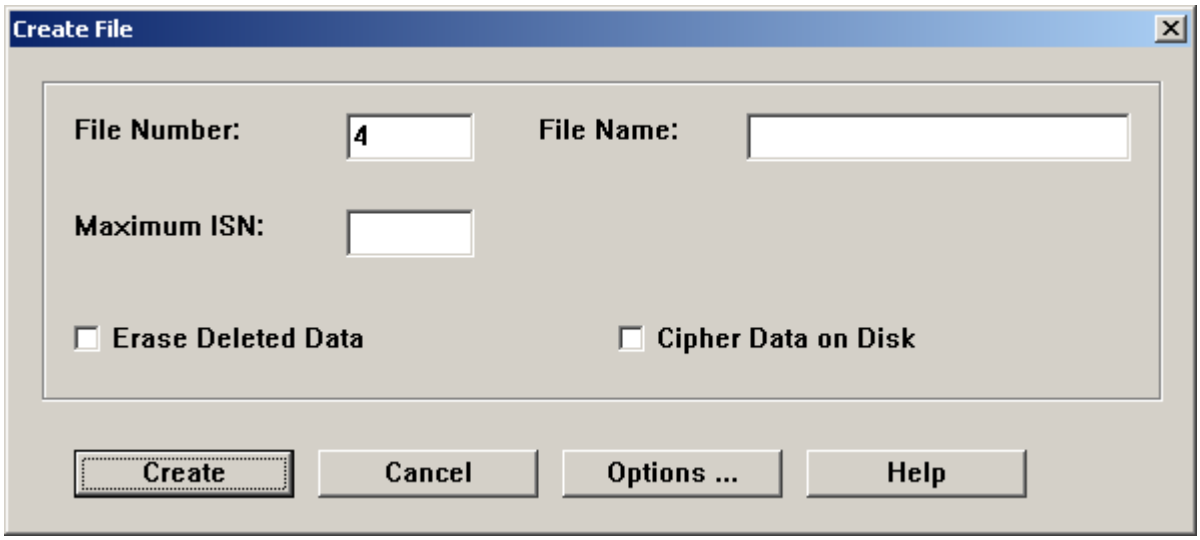


Рис. 5.15. Сохранение значений элементов в структуре файлов

Для успешного определения файла необходимо ввести ряд следующих параметров в данном окне (рис. 5.15.):

Номер файла (File Number)

В данном поле отображается следующий свободный порядковый номер файла. Можно оставить предлагаемой значение, либо ввести иной номер. При этом следует учитывать, что номер не должен превышать максимальное число файлов, определенных для данной БД, а также должен быть отличным от номеров уже существующих файлов и номеров системных файлов, отображаемых в списке файлов БД.

Название файла (File Name)

Название файла, введенное в данное поле, будет отображаться в списке файлов БД и отчетах.

Максимальный ISN (Maximum ISN)

Для дальнейшего определения файла необходимо указать предполагаемый наибольший внутрисистемный номер (Internal Sequence Number - ISN) в текстовом поле ‘Maximum ISN’. Следует учитывать, что от значения данного параметра напрямую зависит размер дискового пространства, которое будет отведено для адресного преобразователя (Address Converter) при создании файла.

Опция “Стереть удаленные данные” (Erase Deleted Data)

Данная опция используется для замещения индексов и блоков хранения данных нулевыми значениями при их возвращении к свободному табличному пространству (удалении).

Опция “Шифровать данные на диске” (Cipher Data on Disk)

Опция используется для включения функции шифрования данных.

Для создания файла после ввода перечисленных параметров следует выбрать ‘Create’. При этом параметры элементов файла и характеристики структуры данных создаваемого файла будут сохранены установленными по умолчанию.

Если все параметры были введены корректно, в списке файлов БД будут отображены номер и имя созданного файла (рис. 5.16).

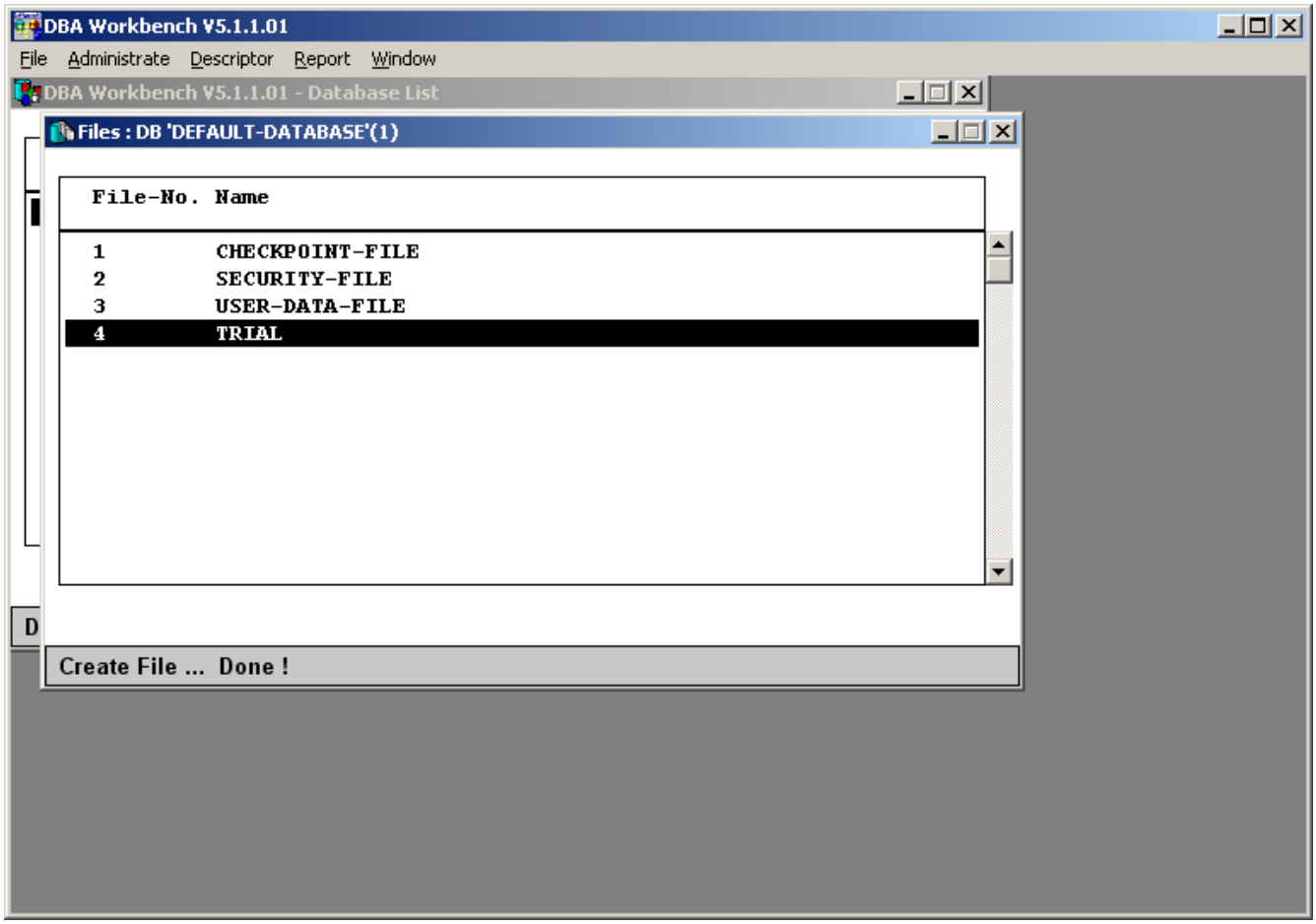


Рис. 5.16. Список файлов БД после создания нового файла БД

4.3. Определение записи

Под записью в данном случае понимается совокупность всех полей, определенных в рамках файла.

Под определением записи понимается последовательное определение каждого из полей (атрибутов) посредством редактора FDT при определении

файла. Каждое из полей (атрибут) определяется рядом обязательных параметров, в то время как часть параметров назначается по выбору.

При определении файла в рамках существующей БД, проектировщик БД вызывает редактор FDT (рис. 5.17.).

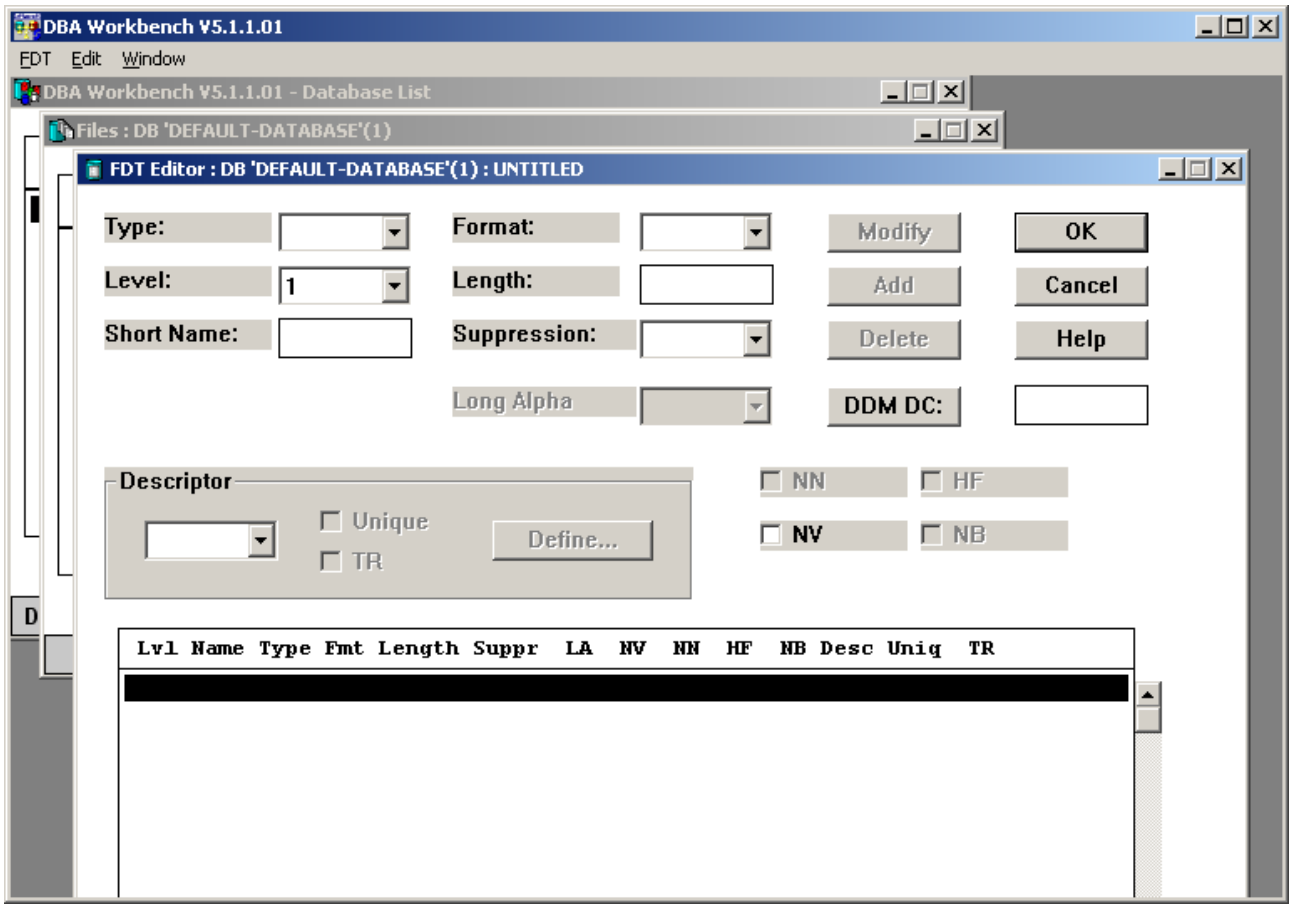


Рис. 5.17. Создание записи при помощи редактора FDT

Далее для каждого поля (атрибута) при помощи текстовых полей и полей со списком вводятся значения следующих параметров.

Тип (Type)

Допускается пустое значение. Следует из списка поля ‘Type’ выбрать необходимый тип определяемого атрибута. По умолчанию системой предлагается пустое значение. Обозначениям данного параметра соответствуют следующие значения (рис 5.18.):

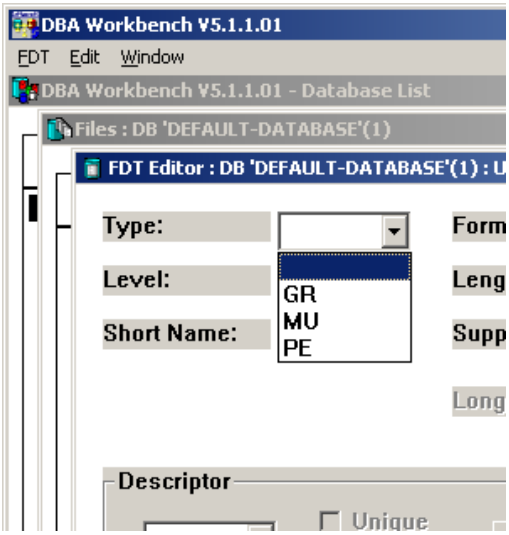


Рис 5.18. Тип определяемого атрибута

пустое значение – простое (элементарное) поле;

GR – группа;

MU – множественное поле;

PE – периодическая группа.

MU: Множественное поле

Опция MU указывает, что поле может содержать больше одного значения в отдельной записи. Фактическое количество значений, присутствующих в каждой записи, может меняться от 0 до 191, но, по крайней мере, одно значение (даже пустое) должно присутствовать в каждой входной записи.

Значения хранятся согласно другим опциям, указанным для поля. Первое значение - это предшествующий полю счетчик, который указывает количество значений, в настоящее время присутствующих в поле. Количество хранимых значений, равное количеству значений, представленных во входной записи, плюс любые значения, добавленные во время обновления поля, меньше любых подавленных значений (это применяется только, если поле определено с опцией NU).

Если количество значений, содержащихся в каждой входной записи постоянно, количество может быть определено в описании оператора MU в

формате MU(n), где "n" равняется количеству значений, представленных в каждой входной записи. Например:

FNDEF='01,AA,5,A,MU(3)'

— показывает, что три значения множественного поля AA присутствуют в каждой входной записи. Значение ноль (0) указывает, что во входной записи никакие значения для множественного поля не присутствуют.

Если количество значений не постоянно для всех входных записей, то однобайтовый двоичный счетчик поля должен предшествовать первому значению каждой входной записи, указывая количество существующих в ней значений.

Все значения, задаваемые во время входа или обновления, будут сжаты (если только не была указана опция FI). Необходимо проявить осторожность при совместном использовании опций FI и MU, если присутствует большое количество сжимаемых значений, то можно потратить впустую большое количество памяти на диске.

Если опция NU определена с опцией MU, то нулевые значения будут подавлены и логически и физически. Позиционность всех значений (включая пустые значения) поддерживается в MU реализации, если не определена опция NU. Если большое количество нулевых значений присутствует в группе поля MU, опция NU может уменьшать требования дисковой памяти для поля, но не должна использоваться, если должны поддерживаться относительные позиции значений.

Опция NC (или NC/NN) не может быть определена для поля с опцией MU.

PE: Периодическая группа

Опция PE указывает, что должна быть определена периодическая группа. Периодическая группа может состоять из одного или более полей и в пределах данной записи могут встречаться от 0 до 99 реализаций (или 191, если определен параметр утилиты ADACMP MAXPE191), но, по крайней

мере, одна реализация (даже, если она содержит все нулевые значения) должна присутствовать в каждой входной записи.

Периодическая группа должна быть описана на уровне 01. Все поля, которые должны содержаться в периодической группе, должны следовать тотчас после и должны быть описаны на уровне 02 или выше (с приращением 1 до максимального 7 уровня). Следующее описание уровня 01 указывает на конец текущей периодической группы.

Опция PE может быть описана только с именем группы. Параметры длины и формата не могут быть определены с именем группы. Периодическая группа может содержать дескрипторы и/или множественные поля и другие группы, но не может содержать другую периодическую группу. Далее следуют примеры:

```
FNOEF='01,GA,PE'                    периодическая группа GA
FNDEF='02.A1,6,A,NU'
FNDEF='02,A2,2,B,NU'
FNDEF='02,A3,4,P,NU'
FNDEF='01,GB,PE(3)'                 периодическая группа GB
FNOEF='02,B1,4,A,DE,NU'
FNDEF='02,B2,5,A,MU(2),NU'
FNDEF='O2,B3'
FNDEF='03,B4,20,A,NU'
FNDEF='03,B5,7,U,NU'
```

Номер уровня (Level)

Не допускается пустое значение. Значение, устанавливаемое по умолчанию – 1. Область допустимых значений – (1-7). Ввести необходимое значение уровня можно либо с помощью клавиатуры, либо выбрав соответствующий уровень из списка. Номера уровней нельзя пропускать, когда назначаются номера уровней для вложенных групп.

Номер уровня - это число, состоящее из одной или двух цифр в диапазоне 01-07 (лидирующие нули необязательны), используемое для построения группы полей. Поля, имеющие номер уровня 02 или больше, входят в состав предшествующей группы, которой назначен более низкий номер уровня.

Описание группы позволяет ссылку на ряд полей (может быть только 1 поле), используя имя группы. Это обеспечивает удобный и эффективный метод ссылки на ряд логических полей.

Для определения группы могут использоваться номера уровней 01-06. Группа может состоять из других групп. Номера уровней нельзя пропускать, когда назначаются номера уровней для вложенных групп.

FNDEF='01,GA' группа

FNDEF='02,A1,...' элементарное или множественное поле

FNDEF='02,A2,...' элементарное или множественное поле

FNDEF='01,GB' группа

FNDEF='02,B1,...' элементарное или множественное поле

FNDEF='02,GC' группа (вложенная)

FNDEF='03,C1,...' элементарное или множественное поле

FNDEF='03,C2,...' элементарное или множественное поле

Поля A1 и A2 находятся в группе GA. Поле B1 и группа GC (состоящая из полей C1 и C2) находятся в группе GB.

Сокращенное имя (Short Name)

Не допускается пустое значение. По умолчанию значение не устанавливается. Имя должно быть уникальным в пределах файла, а также имя должно быть двух символьным. Первый символ – обязательно алфавитный, второй символ может быть как символьным, так и числовым. Специальные символы не допускаются. Также нельзя использовать значения E0-E9, т.к. они зарезервированы системой в качестве редактирующих масок.

Правильные имена: AA, B4, S3, WM

Неверные имена:

A (один символ)

E3 (маска редактирования)

F* (присутствует специальный символ)

6M (первый символ цифра)

Формат (Format)

Не допускается пустое значение. По умолчанию значение не устанавливается. В поле со списком (рис. 5.19.) предлагаются следующие значения параметра Формат:

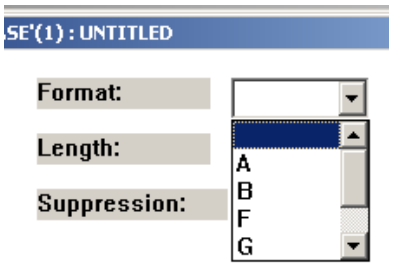


Рис. 5.19. Формат определяемого элемента

A - алфавитно-цифровой (выровненный влево);

B - двоичный (выровненный вправо, без знака/положительный);

F - с фиксированной точкой (выровненный вправо, со знаком, с использованием двойного дополнения для отрицательных чисел);

G - с плавающей точкой (нормализованная форма, со знаком);

P - упакованный десятичный (выровненный вправо, со знаком);

U - распакованный десятичный (выровненный вправо, со знаком).

Стандартный формат используется для того, чтобы определить стандартный формат (по умолчанию), который использует ADABAS во время обработки команды. Указанный стандартный формат вводится в таблицу описания полей и используется при чтении/обновлении поля, до тех пор, пока пользователь не определит значение, перекрывающее стандартный формат.

Стандартный формат определяется для поля. Он не может быть определен для имени группы. Когда группа читается (записывается), поля в пределах группы всегда возвращаются (должно быть обеспечено) согласно стандартному формату каждого индивидуального поля. Указанный формат определяет тип сжатия, которое будет выполнено для данного поля.

Поле с фиксированной точкой имеет длину два или четыре байта. Положительное значение находится в нормальной форме, а отрицательное значение в форме двойного дополнения.

Формат поля с плавающей точкой может быть длиной четыре байта (одноразрядное) или восемь байт (двухразрядное). Поддерживается преобразование значения поля, определенного с плавающей точкой, в другой формат.

Если двоичное поле должно быть дескриптором и поле может содержать положительные и отрицательные числа, то вместо формата "B" должен использоваться формат "F", потому что формат "B" принимает эти значения без знака (как положительные).

Длина (Length)

Допускается пустое значение для имени группы. При определении простого поля указание длины обязательно. По умолчанию значение не устанавливается.

Значение длины используется для того, чтобы определить стандартную длину (по умолчанию), которую использует ADABAS во время обработки команды. Указанная стандартная длина вводится в таблицу описания полей (FDT) и используется при чтении/обновлении поля, до тех пор, пока пользователь не определит значение, перекрывающее стандартную длину.

Максимальные длины полей, которые могут быть определены, зависят от "формата":

Формат	***Максимальная длина***
Алфавитно-цифровой (A)	253 байта

Двоичный (B)	126 байтов
С фиксированной точкой (F)	4 байта (всегда 2 или 4 байта)
С плавающей точкой (G)	8 байтов (всегда 4 или 8 байтов)
Упакованный десятичный (P)	15 байтов
Распакованный десятичный (U)	29 байтов

Для имени группы не может быть определена стандартная длина.

Стандартная длина не ограничивает размер любого данного значения поля, если только не используется опция FI. Команды чтения или добавления могут перекрывать стандартную длину поля, до максимальной длины, разрешенной для этого формата.

Если стандартная длина поля нулевая, то поле принимается как поле переменной длины. Поля переменной длины не имеют стандартной (по умолчанию) длины. В этом случае длина поля будет принимать значения: для полей с фиксированной точкой (F) - только два или четыре байта; для полей с плавающей точкой (G) - только четыре или восемь байтов.

Если во время выполнения команды ADABAS производится обращение к полю переменной длины без указания длины, то будет возвращено значение этого поля, которому предшествует однобайтовая двоичная длина поля (включая байт длины непосредственно). Это значение длины должно быть определено при обновлении поля и также во входных записях. Если поле определено с опцией длинное алфавитно-цифровое (long alpha - LA), значению предшествует двухбайтная двоичная длина поля (включая два байта длины).

Подавление (Suppression)

Допускается и устанавливается по умолчанию пустое значение. Параметры данного поля определяют системные функции подавления нулевых и прочих значений при вводе записи в созданную БД. В поле со списком (рис. 5.20.) предлагаются следующие значения параметра:

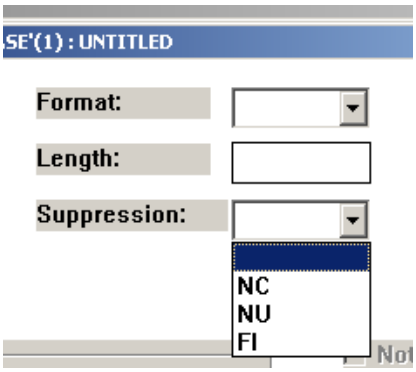


Рис. 5.20. Функция подавления

пустое значение – подавление по умолчанию;

FI – фиксированная длина пространства памяти;

NU – подавление нулевого значения;

NC – SQL-опция нулевого значения.

FI: Фиксированная длина пространства памяти

Опция FI указывает, что поле должно иметь фиксированную длину пространства памяти. Значения в поле хранятся без внутреннего байта длины, не сжаты и не могут быть длиннее, чем заданная длина поля.

Опция FI рекомендуется для полей длиной в один или два байта, которые имеют низкую вероятность содержания пустого значения (количество персонала, пол, и т.д.) и для полей, содержащих значения, которые не могут быть сжаты.

Опция FI не рекомендуется для множественных полей или для полей в пределах периодической группы. Любые пустые значения для таких полей не подавляются (или сжимаются), что может вызвать значительные затраты дисковой памяти и увеличение времени обработки.

Опция FI не может быть определена для:

- полей U-формата;
- полей с опций NC, NN или NU;
- полей переменной длины, определенных с нулевой длиной (0);
- дескрипторов внутри периодической (PE) группы.

Поле, определенное с опцией FI, не может быть обновлено новым значением, которое превышает стандартную длину поля.

Пример использования опции FI (табл. 5.1.):

Пример использования опции FI

Таблица 5.1.

	Описание	**Данные пользователя**	**Внутреннее представление**
Без опции FI	FNDEF='01,AA,3,P'	33104C 00003C	0433104F (4 6айта) 023F (2 байта)
С опцией FI	FNDEF='01,AA,3,P,FI’	33104C 00003C	33104F (3 байта) 00003F (3 байта)

NU: Подавление нулевого значения

Опция NU подавляет нулевые значения, встречающиеся в поле.

Нормальное сжатие (опции NU или FI не определены) приводит к представлению нулевого значения двумя байтами (один байт для значения, длины, второй - непосредственно для значения, в этом случае нулевого значения). Применение опции подавления нулевой величины приводит к внутреннему представлению нулевой величины индикатором однобайтного "нулевого поля". Само нулевое значение не хранится.

Ряд последовательных полей, содержащих нулевые значения, и определенных с опцией NU, представляются однобайтовым "пустым полем" индикатором (двоичное представление 11nnnnnn), где "nnnnnn" - количество соседних байт внутри поля, содержащих нулевые значения, общим числом до 63. По этой причине поля, определенные с опцией NU, должны быть по возможности сгруппированы вместе.

Если опция NU определена для дескриптора, то все нулевые значения для этого дескриптора не хранятся в инвертированном списке. Поэтому результатом команды FIND (редактор Natural), в которой используется этот дескриптор и, для которой нулевое значение используется для поиска, всегда будет отсутствие выбранных записей. Даже в том случае, если в памяти данных имеются записи, которые содержат нулевое значение для

дескриптора. Если дескриптор, определенный с опцией NU, используется для управления логической последовательностью в команде чтения в логической последовательности (L3/L6), записи, которые содержат нулевое значение для дескриптора, не будет читаться.

Дескрипторы, которые нужно использовать в качестве основы для связи файла, и, для которого существует большое количество пустых значений, должен быть определен с опцией NU, чтобы уменьшить общий размер списков связи.

Опция NU не может быть определена для полей, определенных с объединенными опциями NC/NN или с опцией FI.

Пример использования опции NU (табл. 5.2.) - C1 указывает на 1 пустое поле:

Пример использования опции NU

Таблица 5.2.

	Описание	**Данные пользователя**	**Внутреннее представление**
Нормальное сжатие	FNDEF='01,AA,2,B'	0000	0200 (2 байта)
С опцией FI	FNDEF='01,AA,2,B,	0000	0000 (2 байта)
С опцией NU	FNDEF='01,AA,2,B,	0000	C1 (1 байт)

NC: SQL опция нулевого значения

В поле, которое не определено с опцией NC (не подсчитывается), нулевое значение является или нулем или пробелом, в зависимости от формата поля.

В поле, которое определено с опцией NC, нули или пробелы, определенные в буфере записи, интерпретируется по-разному в зависимости от значения "нулевого индикатора": или как истинные нули или пробелы (то есть, как "значащие" нули) или как неопределенные значения (то есть, как истинный SQL или "незначащие" нули).

Если поле, определено с опцией NC, не имеет никакого значения указанного в буфере записи, значение поля всегда обрабатывается, как SQL нулевое значение.

Когда интерпретируется как истинный SQL нуль, нулевое значение удовлетворяет SQL интерпретацию поля, не имеющего никакого значения. Это означает, что значение поля не было введено; то есть, значение поля не определено.

Значение нулевого индикатора ответственно за внутреннее ADABAS представление нуля. Это значение всегда имеет длину два байта и формат с фиксированной точкой, независимо от формата данных. Он определяется в буфере записи, когда значение поля добавляется или обновляется, и возвращается в буфер записи, когда значение поля читается.

Для команд Update (обновления) (Ax) или Add (добавления) (Nx), значение нулевого индикатора должно быть установлено в буфере записи в положение, которое соответствует обозначению полей в форматном буфере. Установка (шестнадцатеричное значение) должна быть одной из следующих:

FFFF - значение поля установлено в "не определено", незначащий нуль; различия между нет значения, двоичные нули или пробелы для поля в буфере записи игнорируются; все интерпретируются одинаково как "нет значения";

0000 - нет значения, двоичные нули или пробелы для поля в буфере записи интерпретируются как незначащие нулевые значения.

Для команд Read (чтения) (Lx) или Find с Read (поиска с чтением) (Sx с форматным буфером), ваша программа должна проверять значение нулевого индикатора, возвращенного в позицию буфера записи, соответствующую позиции поля в форматном буфере. Значение нулевого индикатора - это одно из следующих значений, показывающих содержание фактического значения, которое содержит выбранное поле (шестнадцатеричное значение):

FFFF - ноль или пробел в поле не важны;

0000 - ноль или пробел в поле - значащее значение, то есть истинный ноль или пробел;

xxxx - поле усечено, значение нулевого индикатора содержит длину (xxxx) полного значения, которое хранится в записи базы данных.

Следующий пример (табл. 5.3.) показывает, как нуль представлен двухбайтным двоичным ADABAS полем AA, определенным с опцией NC, для следующего описания поля - **01**,**AA,2,B,NC**:

Представление нуля двухбайтовым полем, определенным с опцией NC

Таблица 5.3.

	Значение нулевого индикатора	**Данные**	**Внутреннее представление ADABAS**
ненулевое значение	0 (значащее двоичное значение)	005	0205
пробел	0 (значащий двоичный ноль)	0000 (ноль)	0200
нуль	FFFF (не значащий двоичный ноль)	(не имеет отношение к данному вопросу)	C1

‘Длинные значения’ (Long Alpha)

Предлагаются следующие опции для данного параметра (рис.5.21.).

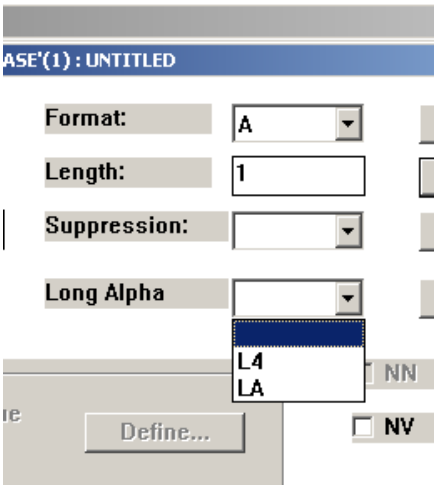


Рис. 5.21. Функция Long Alpha

L4 – длинное алфавитно-цифровое поле (4 байта);

LA - длинное алфавитно-цифровое поле (2 байта).

Опции L4 и LA могут использоваться для полей алфавитно-цифрового формата. При этом такое поле может содержать значение длинной до 16,381 байт.

Дескриптор (Descriptor)

Допускается и устанавливается по умолчанию пустое значение. Любое поле при определении FDT может быть определено дескриптором. Для поля, определенного дескриптором будет сделан вход в инвертированном списке ассоциатора, который позволит этому полю: использоваться в выражении поиска и в качестве ключа сортировки в команде FIND, управлять последовательным логическим чтением или использоваться для связи файла.

Если дескрипторное поле не инициализировано и логически падает мимо конца физической записи, вход инвертированного списка для той записи не формируется по причине производительности. Опция дескриптора должна использоваться осторожно, особенно, если файл большой и поле, которое рассматривается как дескриптор, часто обновляется.

В процессе определения записи предлагаются следующие опции для параметра “Дескриптор” (рис. 5.22.):

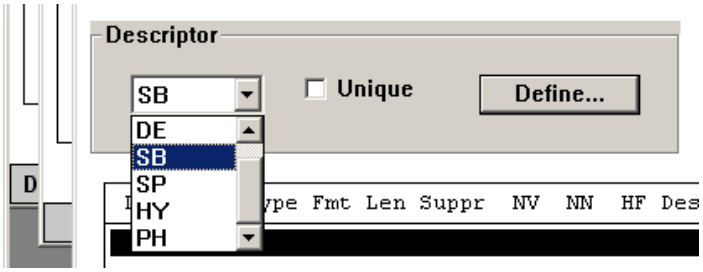


Рис 5.22. Определение дескриптора

DE – дескриптор;

SB – субдескриптор;

SP – супердескриптор;

HY – гипердескриптор;

PH – фонетический дескриптор.

Также при определении всех типов дескриптора (кроме фонетического) может быть использована опция ***уникального дескриптора (Unique)***, а также для определения субдескрипора, супердескриптора, гипердескриптора и фонетического дескриптора используется альтернатива ***Define…*** (рис. 5.22.).

SB: Описание субдескриптора

Субдескриптор - это дескриптор, созданный из части элементарного поля. Элементарное поле может быть или не быть дескриптором непосредственно. Субдескриптор может также использоваться, как субполе; то есть, он может быть определен в форматном буфере для определения формата выходных записей.

Субдескриптор должен быть определен со следующими параметрами:

Имя - имя субдескриптора. Правила присвоения имени субдескриптору, идентичны тем, которые используются для имени поля ADABAS.

UQ (опция Unique) - указывает, что субдескриптор должен быть определен как уникальный.

Исходное поле - имя исходного поля, из которого будет получен субдескриптор.

Начало - относительная позиция байта в пределах исходного поля, где начинается описание субдескриптора.

Конец - относительная позиция байта в пределах исходного поля, где кончается описание субдескриптора.

Подсчет параметров ***начало*** и ***конец*** делается слева направо, начиная с 1 для алфавитно-цифровых полей, и справа налево, начиная с 1 для цифровых и двоичных полей. Если исходное поле определено с форматом P, знак результирующего значения субдескриптора будет взят из 4 битов младшего разряда байта младшего разряда (т. е. 1 байт).

Исходное поле может быть:

- дескриптором;
- элементарным полем;

- множественным полем (но не конкретная реализация множественного поля);
- входить в состав периодической группы (но не конкретная реализация

периодической труппы).

Исходное поле не может быть:

- суб/супер полем, субдескриптором, супердескриптором или фонетическим дескриптором;
- G формата (с плавающей точкой).

Если поле определено с опцией NU, никакие входы не генерируются в инвертированном списке субдескриптора для тех записей, которые содержат нулевое значение (в пределах определенных байт позиций) для поля. Формат тот же самый, что и для исходного поля.

Если исходное поле не инициализировано и логически падает мимо конца физической записи, вход инвертированного списка для той записи не формируется по причине производительности. В этом случае для создания входа инвертированного списка, необходимо файл разгрузить с опцией SHORT, разуплотнить и повторно загрузить файл; или использовать прикладную программу для инициализации поля каждой записи файла.

Синтаксис субдескриптора должен быть описан посредством использования альтернативы ‘Define’ (рис. 5.23.)

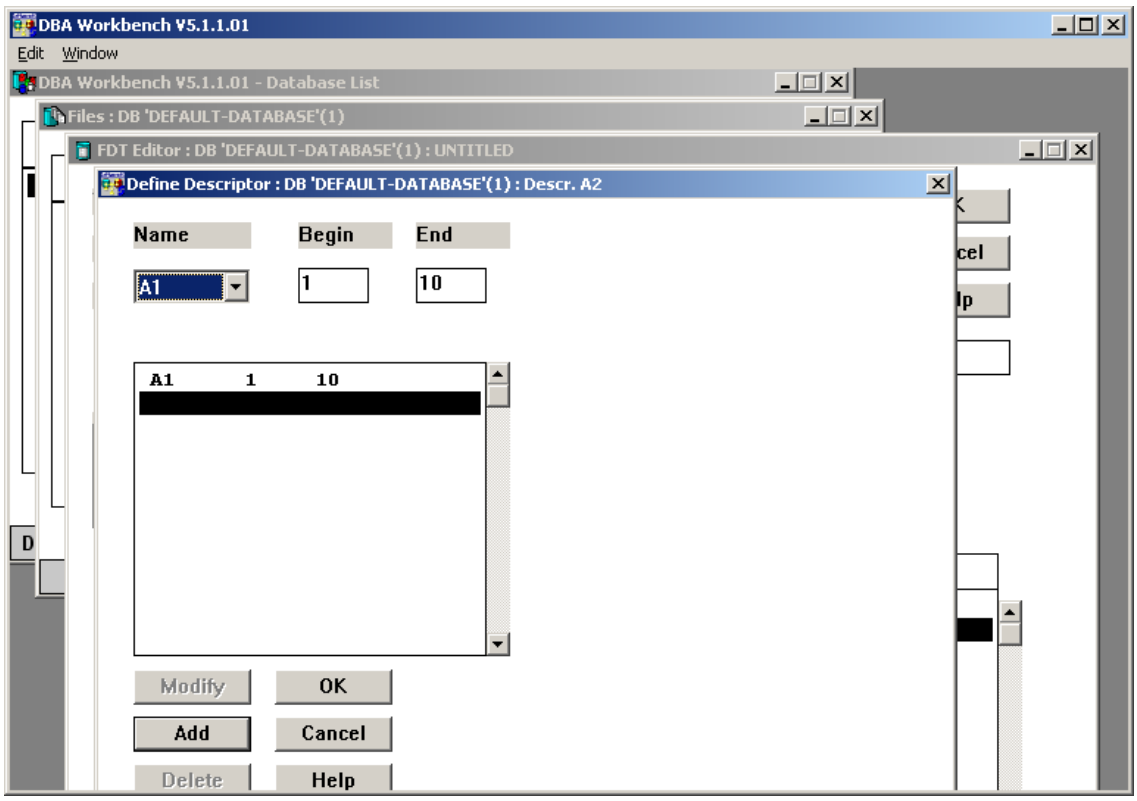


Рис. 5.23. Определение синтаксиса субдескриптора

В окне ‘Define Descriptor’ (рис. 5.23.) необходимо определить ***Исходное поле, начало и конец*** субдескриптора:

Имя исходного поля (Name)

Посредством поля со списком ‘Name’ (отображаются уже определенные поля) необходимо выбрать исходное поле для субдескриптора.

Начало (Begin)

В поле ‘Begin' указывается параметр ***начало*** субдескриптора.

Конец (End)

В поле ‘End' указывается параметр ***конец*** субдескриптора.

Альтернатива ‘добавить’ (Add)

После определения параметров ***имя исходного поля***, ***начало*** и ***конец***, для определения субдескриптора необходимо выбрать альтернативу ‘Add’. При этом запись определенного субдескриптора отображается в соответствующем поле (рис. 5.23.).

Альтернатива ‘изменить’ (Modify)

Для изменения указанных параметров субдескриптора следует использовать опцию ‘Modify’.

Альтернатива ‘удалить’ (Delete)

Для удаления ранее указанных параметров субдескриптора следует использовать опцию ‘Delete’.

Для сохранения параметров субдескриптора нужно воспользоваться системной кнопкой ‘OK’.

Замечание! При определении субдескриптора следует учитывать, что в качестве параметра Исходное поле, как следует из свойств субдескриптора, может быть выбрано только одно поле из уже определенных. В противном случае, значение данного параметра не будет принято системой, о чем проектировщик БД будет уведомлен сообщением (рис. 5.24.):

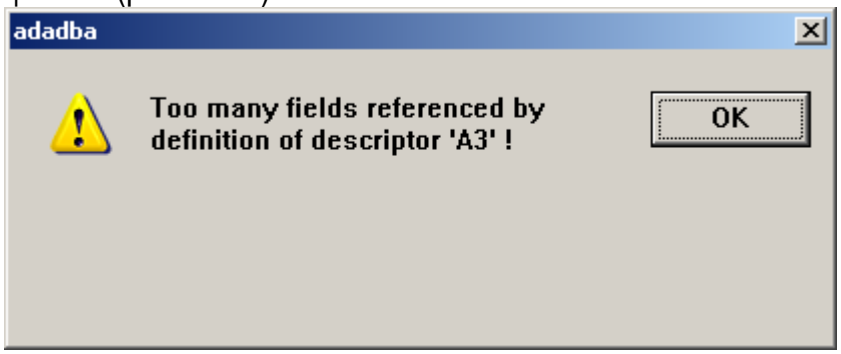


Рис. 5.24. Информационное сообщение, получаемое при выборе нескольких исходных полей

Определение параметров ***Имя*** и ***UQ (Unique)*** субдескриптора осуществляется при помощи основного окна редактора FDT (рис. 5.17.).

SP: Описание супердескриптора

Супердескриптор - это дескриптор, созданный из нескольких полей, частей полей или их комбинации. Каждое поле-первоисточник (или часть поля) используемое для определения супердескриптора, называется исходным. Супердескриптор может быть определен с использованием от 2 до 20 исходных полей или частей поля. Супердескриптор может также быть определен, как уникальный дескриптор. Супердескриптор может также использоваться, как суперполе; то есть, он может быть определен в форматном буфере для определения формата выходной записи.

Субдескриптор должен быть определен со следующими параметрами:

Имя – имя супердескриптора. Правила присвоения имени супердескриптору, идентичны тем, которые используются для имени поля ADABAS.

UQ (опция Unique) - указывает, что супердескриптора должен быть определен как уникальный.

Исходное поле - имя исходного поля, из которого будет получен элемент супердескриптора; может быть определено до 20 исходных полей.

Начало - относительная позиция байта в пределах исходного поля, где начинается описание супердескриптора.

Конец - относительная позиция байта в пределах исходного поля, где заканчивается описание супердескриптора.

Подсчет параметров ***начало*** и ***конец*** делается слева направо, начиная с 1 для алфавитно-цифровых полей, и справа налево, начиная с 1 для цифровых и двоичных полей. Для любого исходного поля кроме тех, которые определены как "FI", значения начала и конца имеют силу в пределах разрешенного диапазона для типа данных исходного поля.

Исходное поле может быть:

- элементарным или максимумом одним множественным полем (но не определенного значения множественного поля);
- в пределах периодической группы (но не определенная реализация);
- дескриптором.

Исходное поле не может быть:

- супер-, суб- или фонетическим дескриптором;
- G формата (с плавающей точкой);
- поле опции NC, если другое исходное поле - поле опции NU;
- длинное алфавитно-цифровое (LA) поле.

Если исходное поле определено с опцией NU, никакие входы не генерируются в инвертированном списке супердескриптора для тех записей, содержащих нулевые значения для поля. Это действительно независимо от

присутствия или отсутствия значений для других элементов супердескриптора.

Если исходное поле не инициализировано и логически падает мимо конца физической записи, вход инвертированного списка для той записи не формируется по причине производительности. В этом случае для создания входа инвертированного списка, необходимо файл разгрузить с опцией SHORT, разуплотнить и повторно загрузить файл; или использовать прикладную программу для инициализации поля каждой записи файла.

Полная длина любого значения супердескриптора не может превышать 253 байта (алфавитно-цифровые) или 126 (двоичных) байтов. Супердескриптор имеет алфавитно-цифровой формат, если какой-нибудь элемент супердескриптора получен из алфавитно-цифрового исходного поля; иначе, супердескриптор имеет двоичный формат.

Все супердескрипторы двоичного формата обрабатываются, как числа без знака.

Синтаксис супердескриптора должен быть описан посредством использования альтернативы ‘Define’ (рис. 5.23.) аналогично описанию синтаксиса субдескриптора.

Замечание! При определении супердескриптора, в отличии от субдескриптора в качестве Исходных полей могут использоваться от 2 до 20 уже определенных в таблице FDT полей.

Определение параметров ***Имя*** и ***UQ (Unique)*** супердескриптора осуществляется при помощи основного окна редактора FDT (рис. 5.17.).

HY: Описание гипердескриптора

Опция гипердескриптора позволяет сгенерировать значения дескриптора на основании алгоритма, обеспеченного пользователем.

Значения основаны на алгоритмах, закодированных в специальных пользовательских выходах гипердескриптора (от HEX01 до HEX31). Каждому гипердескриптору должен быть назначен пользовательский выход,

и отдельный пользовательский выход может обрабатывать множество гипердескрипторов (рис. 5.25.).

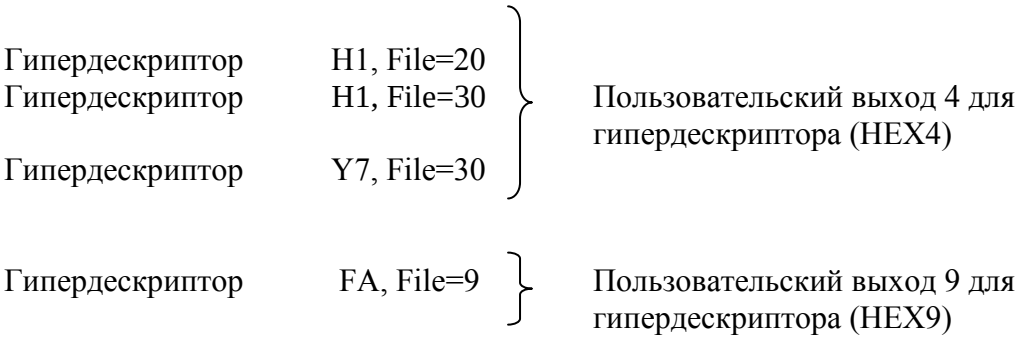


Рис. 5.25. Гипердескриптор и пользовательские выходы

Входные параметры необходимые для пользовательского выхода:

- имя гипердескриптора;
- номер файла;
- адреса полей, взятых из записи памяти данных, вместе с именем поля и PE индексом (если применяется). Эти адреса указывают на сжатые значения полей.

Пользовательский выход должен возвратить значение дескриптора (DVT) в сжатом формате. Может быть не возвращено никакого значения или одно или большее количество значений в зависимости от опций (PE, MU), определенных для гипердескриптора.

Исходный ISN, назначенный для входного значения, может быть изменен.

При использовании гипердескрипторов следует учитывать следующие примечания:

- Проектировщик определяет формат, длину и опции гипердескриптора. Они не могут быть взяты от исходного поля, определенного в параметре HY.
- Поиск, использующий значение гипердескриптора, выполняется тем же способом, что и для стандартных дескрипторов.

- Проектировщик ответствен за создание корректных значений гипердескриптора. Нет никакого стандартного пути проверки значений гипердескриптора на целостность в памяти данных. Пользователь должен установить правила описания каждого значения и проверки корректности значения.
- Если формат гипердескриптора упакованный или распакованный, ADABAS проверяет возвращенные значения на правильность. Знак полбайта для упакованного значения может содержать A, C, E, F (положительный) или B, D (отрицательный). ADABAS преобразует знак в F или D.
- Если файл содержит больше одного гипердескриптора, назначенные выходы вызываются в алфавитном порядке имен гипердескрипторов.

Гипердескриптор должен быть определен со следующими параметрами:

Номер - номер пользовательского выхода, который будет назначен для гипердескриптора. Ядро ADABAS использует этот номер для определения пользовательского выхода гипердескриптора, который будет вызван.

Имя - имя, которое используется для гипердескриптора. Правила присвоения имени гипердескриптору идентичны тем, которые используются для имени поля ADABAS.

Длина - длина гипердескриптора по умолчанию.

Формат – формат гипердескриптора:

Формат	***Максимальная длина***
Алфавитно-цифровой (A)	253 байта
Двоичный (B)	126 байтов
С фиксированной точкой (F)	4 байта (всегда 2 или 4 байта)
С плавающей точкой (G)	8 байтов (всегда 4 или 8 байтов)
Упакованный десятичный (P)	15 байтов
Распакованный десятичный (U)	29 байтов

Опция - опция, которая будет назначена гипердескриптору. Следующие опции могут использоваться вместе с гипердескриптором:

MU - множественное поле;

NU - подавление нулевого значения;

PE - поле периодической группы;

UQ - уникальный дескриптор;

SV – создание поиска значений.

Исходное поле - имя исходного поля. Гипердескриптор может иметь от 1 до 20 исходных полей. Имена полей и адреса передаются пользовательскому выходу.

Если исходное поле определено с опцией NU, никакие входы не генерируются в инвертированном списке гипердескриптора для записей, содержащих нулевые значения поля. Это действительно не зависит от присутствия или отсутствия значений для других элементов гипердескриптора.

Если исходное поле не инициализировано и логически оно попадает мимо конца физической записи, то вход инвертированного списка для этой записи не формируется по причине производительности. В этом случае для создания входа инвертированного списка, необходимо файл разгрузить с опцией SHORT, разуплотнить и повторно загрузить файл; или использовать прикладную программу для инициализации поля каждой записи файла.

Синтаксис гипердескриптора должен быть описан посредством использования альтернативы ‘Define’ (рис. 5.26.)

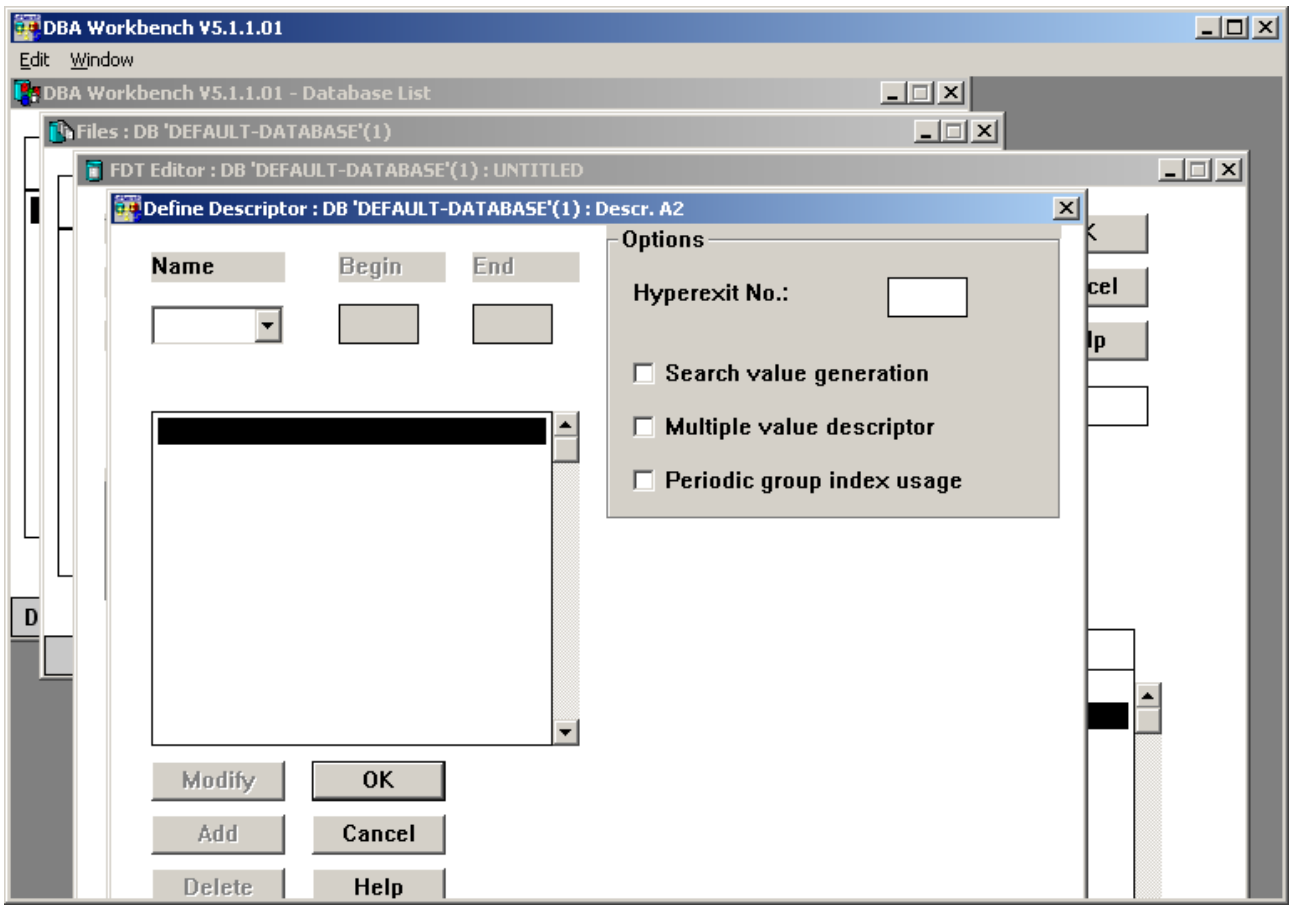


Рис. 5.26. Определение синтаксиса гипердескриптора

В окне ‘Define Descriptor’ (рис. 5.26.) определяются следующие параметры гипердескриптора:

Имя исходного поля (Name)

Посредством поля со списком ‘Name’ (отображаются уже определенные поля) необходимо выбрать ***исходное поле (исходные поля)*** для гипердескриптора.

Начало (Begin) и *Конец (End)*

В полях ‘Begin' и ‘End‘ автоматически устанавливается полная длина ***исходных полей*** гипердескриптора. Для проектировщика эти поля являются неактивными и не подлежат изменению.

Номер пользовательского выхода (Hyperexit No.)

Посредством данного поля необходимо установить параметр ***номер*** гипердескриптора.

Опции (Options)

Здесь определяется такие составляющие параметра ***опция*** гипердескриптора, как MU – альтернатива ‘Multiple value descriptor’, PE – альтернатива ‘Periodic group index usage’ и SV – альтернатива ‘Search value generation’.

Альтернатива ‘добавить’ (Add)

После определения параметров гипердескриптора, необходимо выбрать альтернативу ‘Add’. При этом запись определенного гипердескриптора отображается в соответствующем поле (рис. 5.26.).

Альтернатива ‘изменить’ (Modify)

Для изменения указанных параметров гипердескриптора следует использовать опцию ‘Modify’.

Альтернатива ‘удалить’ (Delete)

Для удаления ранее указанных параметров гипердескриптора следует использовать опцию ‘Delete’.

Для сохранения параметров гипердескриптора нужно воспользоваться системной кнопкой ‘OK’.

Определение параметров ***имя***, ***длина***, ***формат*** гипердескриптора, а также таких составляющих параметра ***опция***, как UQ и NU осуществляется при помощи основного окна редактора FDT (рис. 5.17.), аналогично определению параметров ***имя***, ***UQ (Unique)*** суб- и супердескриптора.

Для лучшего понимания процедуры определения гипердескриптора ниже приведен пример описания гипердескриптора.

Для этого примера используются следующее описание (табл. 5.4.):

Пример описания гипердескриптора

Таблица 5.4.

Описание определенных полей	**Значения полей**
FNDEF='01,LN,20,A,DE,NU'	Фамилия
FNDEF='01,FN,20,A,MU,NU'	Имя
FNDEF='01,ID,4,B,NU'	Идентификатор
FNDEF='01,AG,3,U'	Возраст
FNDEF='01,AD,PE'	Адрес
FNDEF='02,CI,20,A,NU'	Город
FNDEF='02,ST,20,A,NU'	Улица
FNDEF='01,FA,PE'	Родственники
FNDEF='02,NR,20,A,NU	Фамилия родственника
FNDEF='02,FR,20,A,MU,NU'	Имя родственника

Описание гипердескриптора: **HYPDE=’2,NH,60,A,MU,NU=LN,FN,FR’**

Гипердескриптору назначен пользовательский выход 2 и имя гипердескриптора - HN.

Длина гипердескриптора - 60, формат - алфавитно-цифровой и множественное (MU) поле с подавлением нулей (NU).

Значения для гипердескриптора должны быть получены из полей LN, FN и FR (см. табл. 5.4.).

PH: Фонетический дескриптор

В результате выполнения команды FIND (редактор Natural) с использованием фонетического дескриптора, все возвращенные записи будут содержать подобные фонетические значения (иметь аналогичное произношение). Фонетическое значение дескриптора основано на первых 20 байтах значения поля. Рассматриваются только алфавитные значения; числовые значения, специальные символы и пробелы игнорируются. Нижний и верхний регистры алфавитно-цифровых символов внутренне идентичны.

Фонетический дескриптор должен быть определен со следующими параметрами:

Имя - имя, которое будет использоваться для фонетического дескриптора. Правила присвоения имени фонетическому дескриптору, идентичны тем, которые используются для имени поля ADABAS.

Поле - имя поля, которое будет транскрибировано фонетически.

Поле должно быть:

- элементарным или множественным полем;
- определенно в алфавитно-цифровом формате.

Поле может быть дескриптором.

Поле не может:

- быть субдескриптором, супердескриптором или гипердескриптором;
- входить в состав периодической группы;
- использоваться, как исходное поле, для более, чем одного фонетического дескриптора.

Если поле определено с опцией NU, никакие входы не генерируются в инвертированном списке фонетического дескриптора для тех записей, которые содержат нулевое значение (в пределах определенных байт позиций) для поля. Формат тот же самый, что и для поля.

Если поле не инициализировано и логически падает мимо конца физической записи, вход инвертированного списка для той записи не формируется по причине производительности. В этом случае для создания входа инвертированного списка, необходимо файл разгрузить с опцией SHORT, разуплотнить и повторно загрузить файл; или использовать прикладную программу для инициализации поля каждой записи файла.

Синтаксис фонетического дескриптора должен быть описан посредством использования альтернативы ‘Define’ (рис. 5.27.)

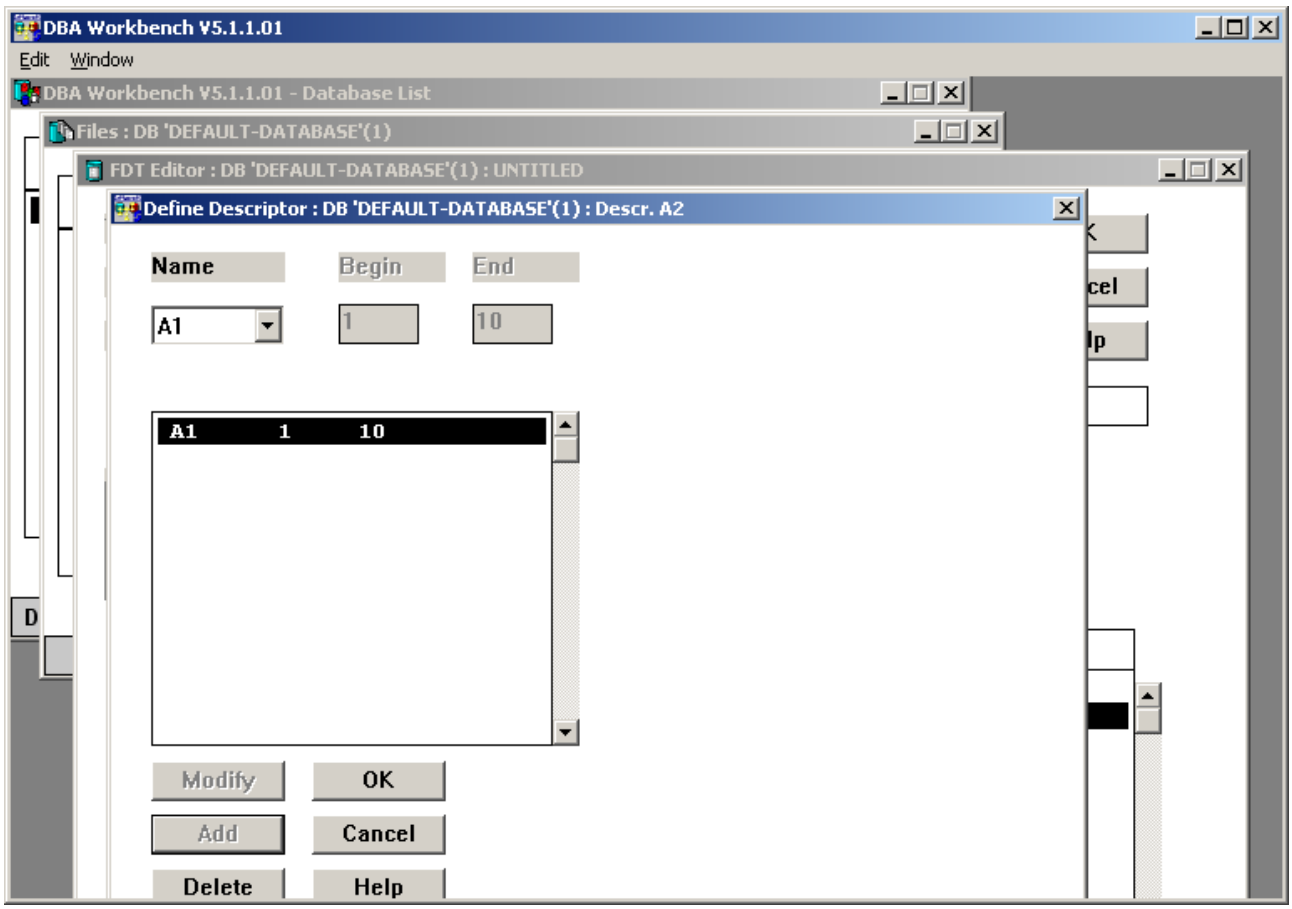


Рис. 5.27. Определение синтаксиса фонетического дескриптора

В окне ‘Define Descriptor’ (рис. 5.27.) определяются следующие параметры фонетического дескриптора:

Имя исходного поля (Name)

Посредством поля со списком ‘Name’ (отображаются уже определенные поля) необходимо выбрать ***поле*** фонетического дескриптора.

Начало (Begin) и *Конец (End)*

В полях ‘Begin' и ‘End‘ автоматически устанавливается полная длина ***поля*** фонетического дескриптора. Для проектировщика эти поля являются неактивными и не подлежат изменению.

Альтернатива ‘добавить’ (Add)

После определения параметров фонетического дескриптора, необходимо выбрать альтернативу ‘Add’. При этом запись определенного фонетического дескриптора отображается в соответствующем поле (рис. 5.27.).

Альтернатива ‘изменить’ (Modify)

Для изменения указанных параметров фонетического дескриптора следует использовать опцию ‘Modify’.

Альтернатива ‘удалить’ (Delete)

Для удаления ранее указанных параметров фонетического дескриптора следует использовать опцию ‘Delete’.

Для сохранения параметров фонетического дескриптора нужно воспользоваться системной кнопкой ‘OK’.

Определение параметра ***имя*** фонетического дескриптора осуществляется при помощи основного окна редактора FDT (рис. 5.17.), аналогично определению параметров ***имя***, ***UQ (Unique)*** суб- и супердескриптора.

Ниже приведен пример описания фонетического дескриптора:

Описание поля: **FNDEF='01,AA,20,A,DE,NU'**

Фонетическое описание: **PHONDE='PA(AA)'**

NN: SQL опция не нулевого значения

При помощи редактора FDT для каждого элементарного поля может быть задана функция NN (Not NULL – рис. 5.17.)

Опция NN ("не ноль" или "нулевое значение не разрешено") может быть определена только, когда опция NC также определена для поля. Опция NN указывает, что NC поле должно всегда иметь определенное значение (включая ноль или пробел); оно не может содержать "нет значения".

Опция NN гарантирует, что поле не будет оставлено неопределенным, когда запись добавлена или обновлена; поле всегда должно устанавливаться в значащее значение.

Следующий пример показывает, как незначащий ноль будет обработан в двухбайтном алфавитно-цифровом ADABAS поле AA, когда оно определено с и без опции NN (табл. 5.5.):

Пример представления нулей с опцией NN

Таблица 5.5.

Опция	Описание поля	Значение нулевого индикатора	Внутреннее представление ADABAS
С функцией	01,AA,2,A,NC.NN	FFFF (незначащие	Нет, ошибка
Без функции NN	01,AA,2,A,NC	FFFF (незначащие нули)	C1

Ниже следует пример корректного описания поля AA с назначением опций NC и NN:

FNDEF='01,AA,4,A,NN,NC,DE'

Далее следует пример неправильного использования опций NC/NN. Эти описания приведут к тому, что в результате будет получена ошибка:

FNDEF='01,AA,4,A,NC,NU' (опции NU и NC несовместимы)

FNDEF='01,AB,4,A,NC,FI' (опции NC и FI не совместимы)

FNDEF='01,PG,PE' (не разрешается использовать опцию NC

FNDEF='02,P1,4,A,NC' внутри PE группы)

Альтернатива ‘добавить’ (Add)

После определения каждого поля следует выбрать альтернативу ‘Add’ в основном окне редактора FDT (рис. 5.17.). При этом системой осуществляется проверка синтаксиса каждого поля. Если системой будут выявлены какие-либо ошибки, проектировщик будет осведомлен посредством сообщений. Для продолжения работы с редактором необходимо исправить все ошибки, выявленные системой.

Альтернатива ‘изменить’ (Modify)

Для внесения изменений для уже введенных параметров полей необходимо выделить (с помощью курсора мыши) нужное поле и воспользоваться альтернативой ‘Modify’ (рис. 5.17.).

Альтернатива ‘удалить’ (Delete)

В случае, когда необходимо удалить какое-либо из определенных полей следует выделить (с помощью курсора мыши) нужное поле и воспользоваться альтернативой ‘Delete’ (рис. 5.17.).

После определения параметров атрибутов следует воспользоваться системной кнопкой ‘OK’ основного окна редактора FDT (рис. 5.17.) и перейти к процедуре сохранения файла.

4.4. Определение параметров ассоциатора и обработки данных

Под определением элементов файла в рамках создания БД подразумевается определение экстентов таких параметров, как ***хранилище данных (Data Storage*** или **DATA*)***, ***адресный конвертор (Address Converter*** или **AC*)***, ***стандартный индекс (Normal Index*** или **NI*)***, ***верхний индекс (Upper Index*** или **UI*)***. Перечисленные параметры используются для указания типа и размера экстента, который будет определен. Экстенты этих параметров вводятся и корректируются при помощи опций при сохранении файла БД.

Для определения экстентов данных параметров необходимо создать файл в рамках существующей БД и определить все поля для внесения пользовательских данных. При сохранении определенного файла (при помощи окна ‘Create File’ – рис. 5.28.) необходимо воспользоваться системной кнопкой ‘Опции’ (‘Options…’).

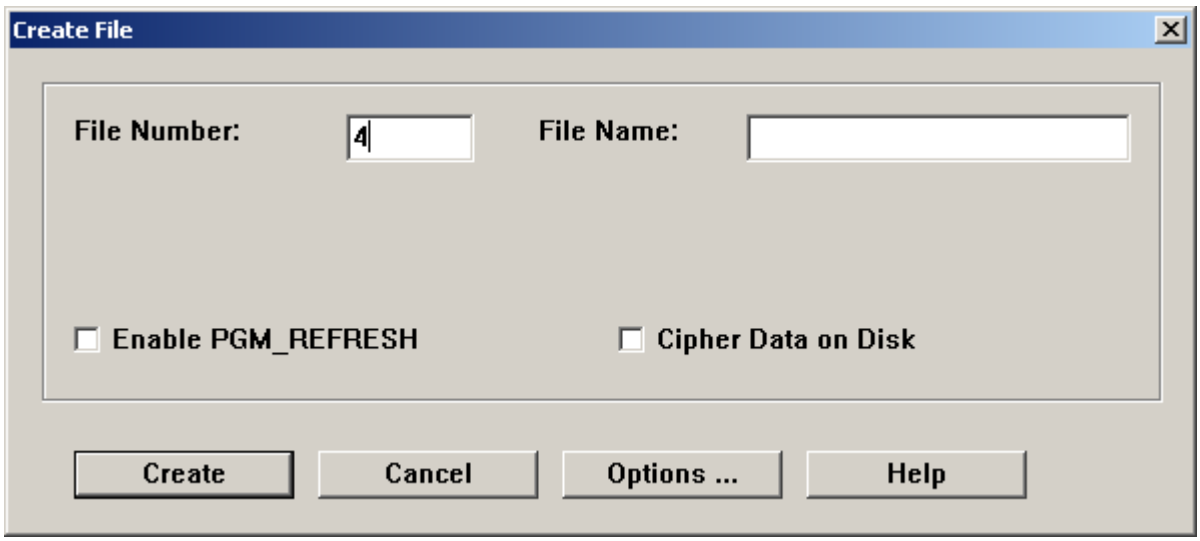


Рис. 5.28. Определение экстентов параметров

Далее проектировщику будет предложено окно для определения параметров (рис.5.29.)

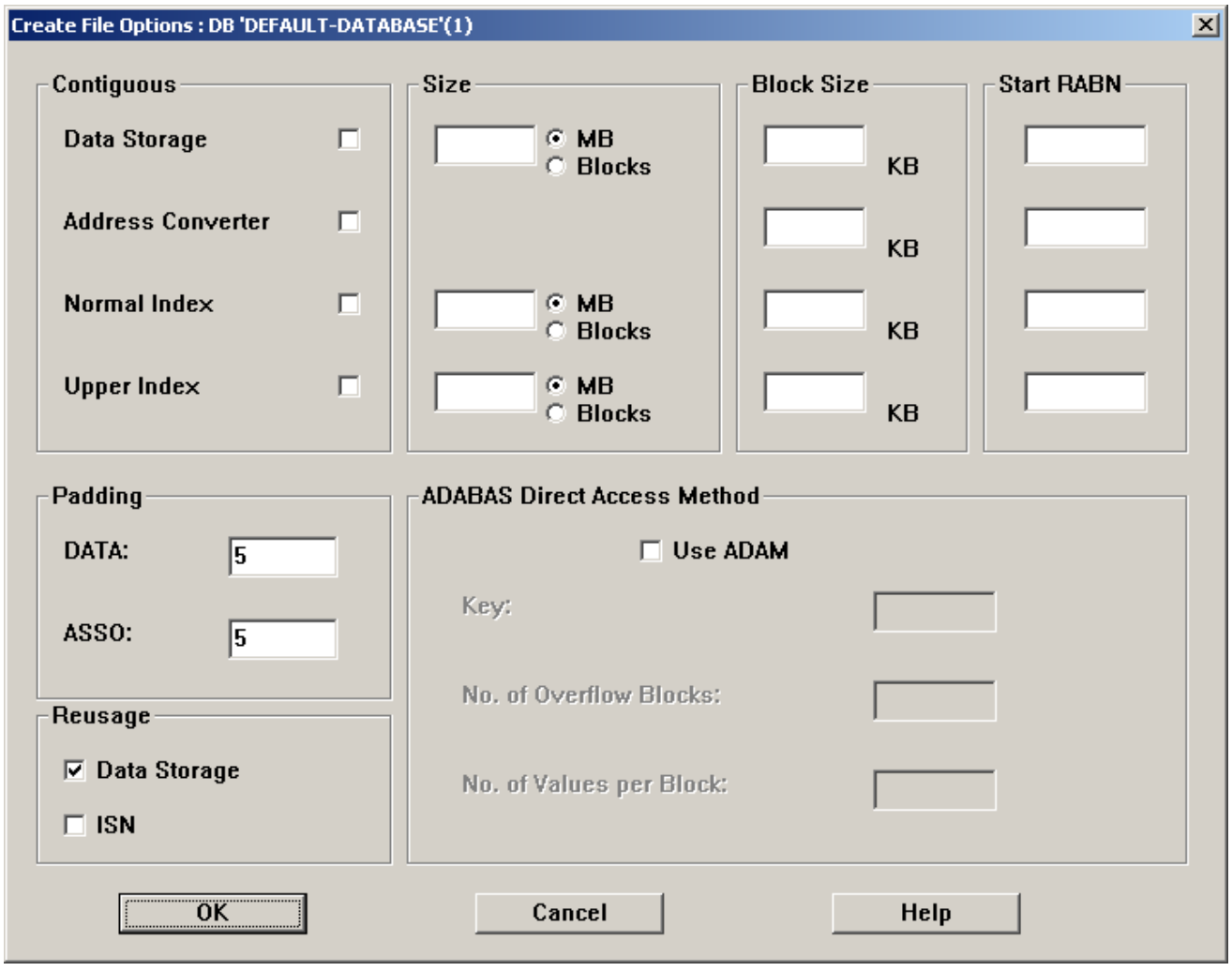


Рис 5.29. Определение параметров

В появившемся окне (рис. 5.29.) предлагаются следующие группы для ввода параметров:

Непрерывные (Contiguous)

Для внесения значений любого из параметров ***хранилище данных***, ***адресный конвертор***, ***стандартный индекс***, ***верхний индекс*** необходимо поставить флажок напротив соответствующего параметра.

Размер (Size)

В данной группе значений указывается размер (в блоках – опция ‘Blocks’ или мегабайтах – опция ‘MB’) каждого из параметров ***хранилище данных,*** ***стандартный индекс***, ***верхний индекс***. Размер адресного конвертора фиксирован.

Размер блока (Block Size)

Для определения размера блоков каждого из параметров необходимо ввести значение (в килобайтах) в соответствующее параметру поле данной группы.

Начальный RABN (Start RABN)

Если необходимо разместить область какого-либо из параметров на определенном дисковом сегменте или в специальном экстенте, следует ввести значение начального RABN в соответствующее параметру поле. В случае, когда начальный RABN не определен, система размещает нижние ряды последовательных свободных RABN так, чтобы обеспечить достаточно свободного дискового пространства для каждого из элементов.

Замечание! Если размеры и начальный RABN не определены проектировщиком, они будут автоматически назначены для каждого из вышеперечисленных параметров.

Дополнительное пространство (Padding)

При определении значений дополнительного пространства для файла ассоциатора (ASSO) и файла данных (DATA), системой резервируется область дополнительного дискового пространства для записей, которые могут потребовать дополнительное место при обновлении. Это позволяет избежать перемещения записей в иные блоки. Если ожидается незначительное (или отсутствие) обновление дескрипторов (файл ASSO) и незначительное (или отсутствие) расширения записей (файл DATA), следует установить значение данного параметра в диапазоне от 0 до 10. Если же ожидается широкое обновление дескрипторов и большое расширение записей, значение данного параметра необходимо определить в диапазоне от 10 до 50.

Повторное использование (Reusage)

Данная опция используется для повторного использования блоков данных и номеров ISN. При использовании опции *хранение данных (Data Storage)* при добавлении новой записи или перемещении записей в результате осуществления обновлений, системой используется первый найденный блок, в котором имеется достаточно пространства для размещения записи. При использовании опции *номер ISN (ISN)*, номера ISN тех записей, которые были удалены перемещаются (присваиваются иным записям), в противном случае (когда данная опция не используется) при добавлении новой записи ей присваивается следующий (наибольший) номер ISN. Особенность использования опции *номер ISN* можно рассмотреть на примере (табл. 5.6.).

Пример использования опции *номер ISN*

Таблица 5.6.

Последовательность ISN до удаления записей и внесения новой записи	**Последовательность ISN при использовании опции *Reusage номер ISN***	**Последовательность ISN без использования опции *Reusage номер ISN***
… **149** **150** **151** **152**	**…** **149** **150** **151**	**…** **149** **152** **153**

В первоначальном состоянии имелось несколько записей, и их номера ISN были расположены по порядку (первая колонка табл. 5.6.). Допустим, что было удалено две записи (соответствующих номерам ISN – 150 и 151), и одна запись была добавлена. Результат присвоения номеров ISN в данном случае при и без использования опции *Reusage номер ISN*, показан во второй и третьей колонках таблицы 5.6.

Метод направленного запроса ADABAS (ADABAS Direct Access Method или ADAM)

Метод направленного запроса ADABAS (ADAM) позволяет поиск записей прямо из ***хранилища данных*** без обращения к инвертированным спискам. Номер блока ***хранилища данных***, в котором располагается запись, высчитывается на основе рандомизированного алгоритма, на основе ADAM-ключа записи. ADAM-ключ каждой записи должен быть уникальным. Номер ISN записи также может использоваться в качестве ADAM-ключа. Для использования опции ADAM должны быть определены следующие параметры (рис. 5.29):

Использовать ADAM (Use ADAM)

При введения параметров для использования в рамках метода ADAM, необходимо поставить выбрать данную опцию. Это делает поля для ввода параметров активными для проектировщика.

ADAM-ключ (Key)

В данном поле указывается ***сокращенное имя (Short Name)*** определенного в файле дескриптора, либо номер ISN. В качестве ADAM-ключа не может использоваться субдескриптор, супердескриптор, фонетический дескриптор или гипердескриптор, а также множественное поле, или поле в рамках заданной периодической группы и поля, определенные с опцией NU/NC.

Замечание! В качестве ADAM-ключа необходимо указывать какой-либо из уникальных дескрипторов, определенных в рамках создаваемого файла, либо номер ISN. В случае, когда объявленный ADAM-ключ не является уникальным дескриптором, проектировщик будет проинформирован сообщением (рис. 5.30.):

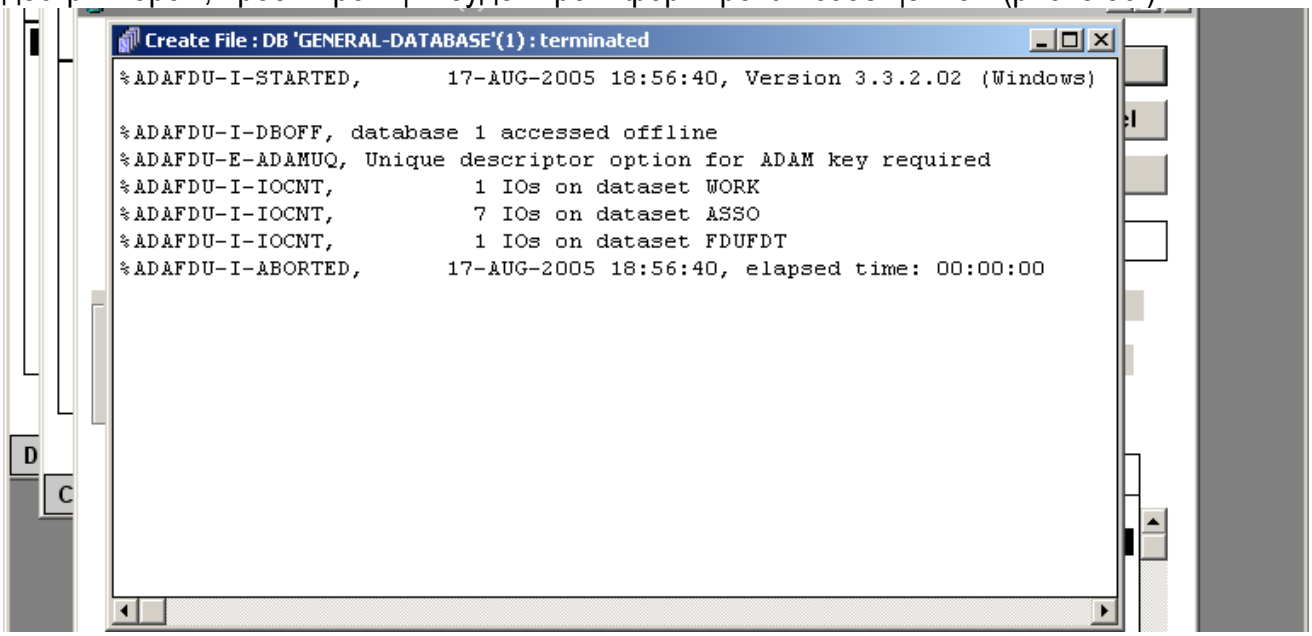


Рис. 5.30. Информационное сообщение, получаемое если ADAM-ключ не является дескриптором

Количество дополнительных блоков (No. Of Overflow Blocks)

При использовании опции ADAM необходимо определить количество блоков, отводимых дополнительно для осуществления операций. Требуется как минимум, один блок. Количество блоков может быть увеличено в дальнейшем. Дополнительные блоки используются, когда недостаточно дискового пространства для посчитанного методом ADAM блока новой записи. Преимущество при выполнении операций с использованием метода ADAM повышается с ростом количества дополнительных блоков.

Количество значений в блоке (No. of Values per Block)

Данный параметр используется для распределения записей данных. Если ADAM-ключ определен как ISN, или дескриптор с фиксированной точкой, данный параметр определяет количество последовательных значений, которые будут храниться в одном блоке.

Если ADAM-ключ является дескриптором формата алфавитно-цифрового, двоичного или с плавающей точкой, данный параметр определяет смещение с конца значения записи на 4 байта (значение, используемое как ключ).

Следующий пример (рис. 5.31.) наглядно показывает действие данного параметра.

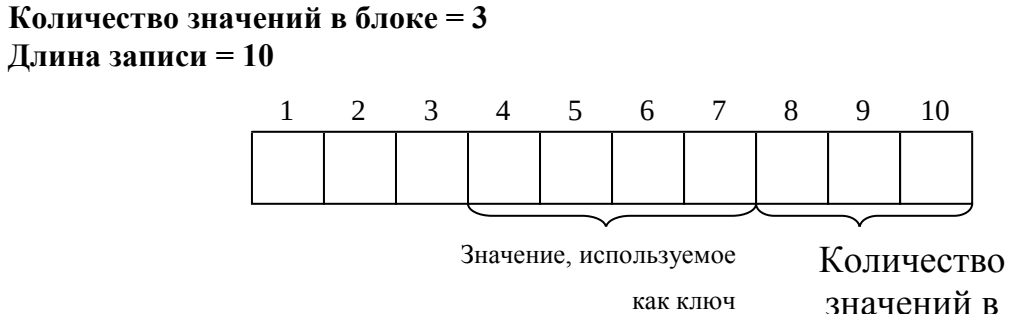


Рис. 5.31. Действие параметра ADAM (формата алфавитно-цифрового, двоичного или с плавающей точкой)

Если длина значение для определения смещения оказывается меньше, чем 4 байта, то это значение будет расширено до нужной длины двоичными нулями.

Если формат ADAM-ключа – запакованный или не запакованный – количество значений в блоке определяет смещение (в байтах) с конца значения записи до той позиции значения, в которой значение может быть рассмотрено в качестве значения ADAM. Это значение ADAM будет преобразовано в 4-х байтовое целочисленное значение. Дальнейший пример иллюстрирует действие функцию данного параметра для запакованного и распакованного формата ADAM-ключа (рис.5.32.).

Количество значений в блоке = 2
Длина записи = 8

Рис. 5.32. Действие параметра ADAM (формата запакованный или не запакованный)

Главное преимущество использования дескрипторов в качестве ADAM-ключа – понижение количества запросов к списку номеров ISN. Главное преимущество использования ISN в качестве ADAM-ключа – понижение количества запросов к ***адресному конвертору***.

После определения всех параметров следует воспользоваться системной кнопкой ‘OK’ (рис. 5.29.) и перейти к сохранению файла.

6. ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЙ КИС ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ В СРЕДЕ NATURAL

6.1. Основные возможности и компоненты редактора Natural

Natural – является одним из немногих средств 4-го поколения, с помощью которого можно создавать комплексные, высокопроизводительные приложения, критичные с точки зрения обеспечения повседневной непрерывной успешной работы предприятий. В нем реализована концепция активных интерфейсов с внешними компонентами как для периода разработки, так и для периода исполнения прикладных программ.

Языки программирования Natural позволяют производить управление множеством СУБД, в т.ч. ADABAS, DB2, DL/1,SQL/DC, VSAM, IMS/DB, ORACLE, SYBASE, RMS.

В состав Natural входят следующие компоненты:

1. Компоненты NATURAL. Для достижения провозглашенных целей в состав NATURAL включены следующие средства: подсхемы данных, интеллектуальные редакторы, единая среда разработки пользователя, управление версиями, утилита переноса приложений, поддержка языка SQL, диалоговый инструментарий отладки, командный процессор, интегрированная система управления.
2. Активные интерфейсы.
3. Среда выполнения. Для основных платформ предоставляются исполнительные системы NATURAL, обеспечивающие выполнение готовых приложений и миграцию приложения с одной платформы на другую.

Natural – единый язык для ADABAS, DB2 и прочих СУБД

Natural впервые был представлен в 1979 году, и к настоящему времени установлен в более чем 3000 корпорациях. Большинство клиентов много лет используют Natural для своих критически важных приложений. На рис. 6.1. показана связь Natural с различными СУБД.

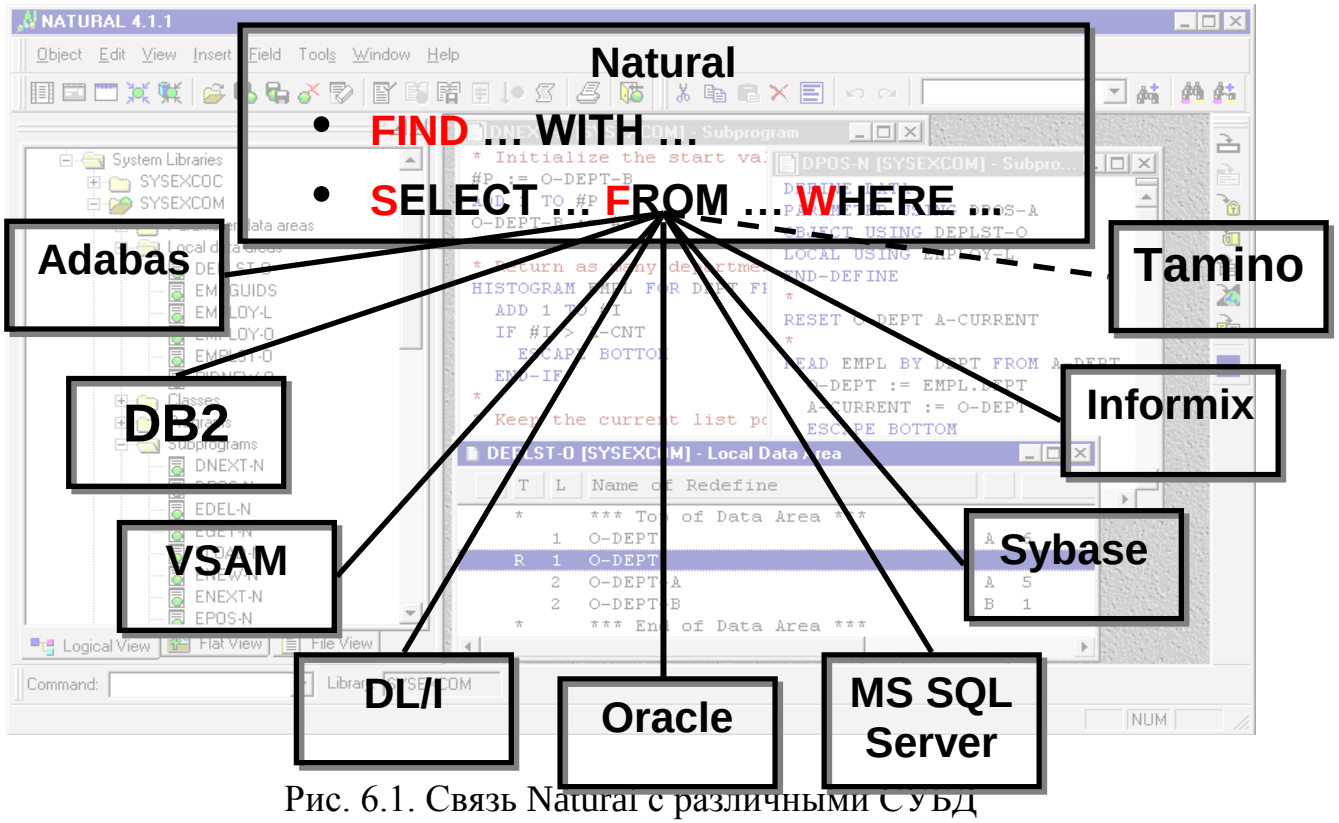


Рис. 6.1. Связь Natural с различными СУБД

Область применения Natural

Natural является языковым средством 4-го поколения, с помощью которого можно создавать комплексные, высокопроизводительные приложения, обеспечивающие автоматизацию бизнес процессов предприятий разных отраслей: промышленность, торговля, транспорт, связь, медицина, банки и страхование, государственное и муниципальное управление, общественные организации и многие другие области человеческой деятельности.

Встроенная система Natural Studio

Специализированные редакторы облегчают работу программистов, что способствует эффективной и удобной разработке и модификации программных объектов Natural. В Natural эти редакторы, средства просмотра библиотек и другие инструменты объединены в интегрированную систему Natural Studio.

Раннее предупреждение об ошибках

Эти принципы проектирования были неотъемлемой частью Natural с самого начала. Возможность раннего предупреждения об ошибкках (при

которой ошибки обнаруживаются при компиляции, а не в процессе выполнения).

Встроенная поддержка баз данных

Natural обеспечивал поддержку баз данных на уровне синтаксиса языка. Таким образом, сглаживались несоответствия между подходом, принятым в системах управления базами данных, ориентированным на множества, и процедурным подходом программных языков. Начав с поддержки высокопроизводительной постреляционной СУБД Adabas компании Software AG, в Natural вскоре была обеспечена прозрачная поддержка других СУБД и индексно-последовательных моделей хранения данных, например, IMS или VSAM. Когда широкую популярность завоевал SQL, в Natural была реализована поддержка SQL на уровне синтаксиса языка, что позволяло компилятору проверять корректность выражений SQL.

Кроме того, Natural поддерживает вызовы хранимых процедур SQL в виде встроенных конструкций языка.

Производительность Natural

Natural разработан для реализации бизнес - приложений, и в течение многих лет совершенствовался в непрерывном диалоге с заказчиками. Мощные высокоуровневые конструкции, такие как FIND (оператор запроса или обращения к базе данных), DECIDE (оператор ветвления или выбора), панели экранов и диалоги (пользовательский интерфейс) позволяют запрограммировать стандартные задачи всего несколькими операторами языка. Интеграция с Predict, гарантирует простоту поддержки программ на Natural и не требует больших трудозатрат на написание первичной документации.

Развитые пользовательские интерфейсы Natural

Графический интерфейс (GUI)- *Natural Maps* (*конструктор панелей экранов*) обеспечивает поддержку традиционной модели, ориентированной

на программы. Natural Maps отличается первоклассными возможностями переносимости и может работать в пакетном режиме и на всех интерактивных устройствах - от классических терминалов 3270 до ПК-клиентов, имеющих такие элементы графического интерфейса как меню, кнопки, поля прокрутки или изображения.

Natural Web Interface является связующим звеном между Web Сервером (HTTP сервер) и средой Natural. Natural Web Interface поддерживает:

- вызов подпрограмм Natural со страницы Web;
- обратную связь пользователя со страницей Web;
- создание Web страниц.

Natural – единый язык разработки на различные OS/Hardware платформы

Natural работает на множестве платформ от мэйнфреймов, OpenVMS, AS/400, UNIX до MS Windows. Исходные коды Natural можно легко перенести с одной платформы на другую, что дает пользователю возможность переходить от одной серверной технологии к другой. На рис.6.2. перечислены основные платформы, с которыми работает Natural.

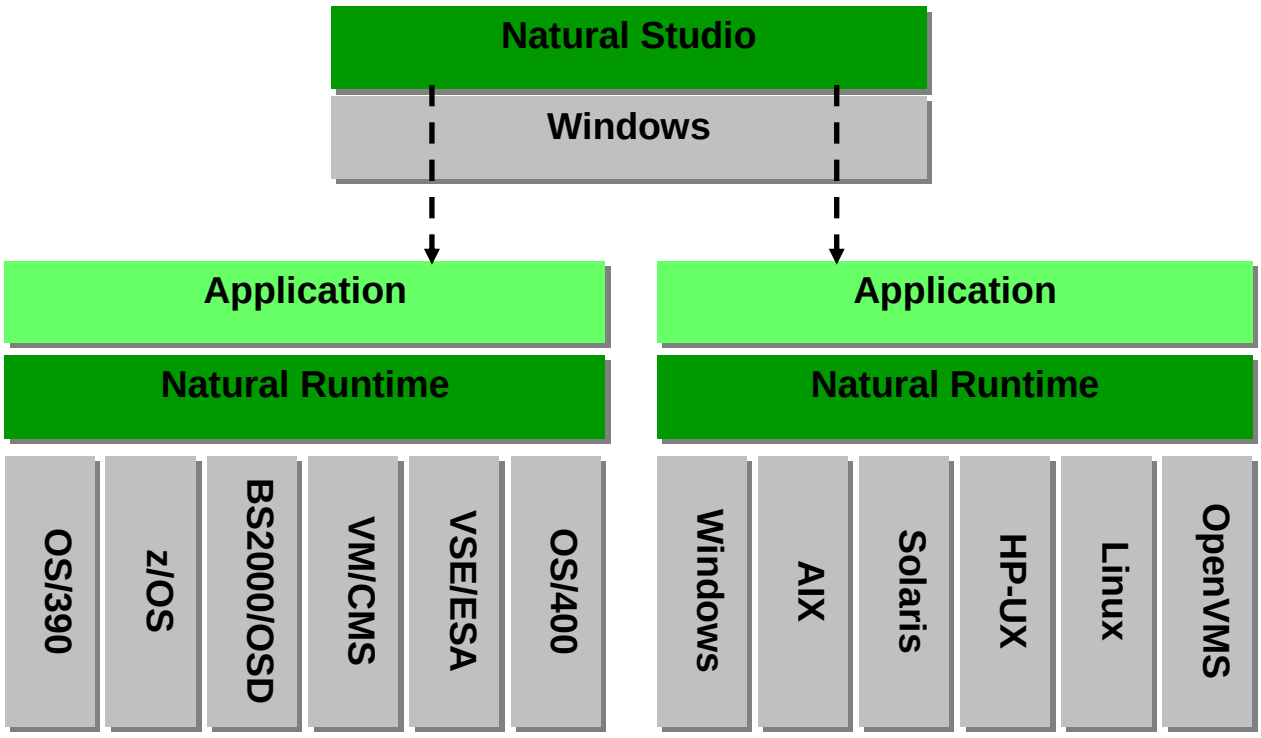


Рис. 6.2. Связь Natural с различными платформами

Интероперабельность

Natural не оторван от других технологий программирования. Он допускает вызов модулей, написанных на таких языках, как Кобол, Си, Java, Visual Basic.

Клиентские программы, написанные не на Natural, могут вызывать методы из компонентов Natural, а Natural- клиенты могут вызывать методы компонентов, созданных не на Natural.

Безопасность

Согласно исследованиям (IDC) около 90% нарушений защиты информационных технологий совершаются сотрудниками предприятий. Другими словами, только в 10% случаев злоумышленники проникают в системы через Internet или путем перехвата данных при передаче. Большинство неавторизованных "посещений" приходится на сервер.

Поэтому заниматься защитой данных и приложений нужно на сервере. Natural Security представляет собой дополнительный пакет обеспечения безопасности для Natural, доступный на всех поддерживаемых платформах.

Он дает пользователям возможность установить четко определенные права доступа к функциям приложений, модулям, библиотекам и глобальным функциям, как для индивидуальных пользователей, так и для их групп. Natural Security хорошо масштабируется; инструкции по защите можно легко перенести вместе с приложением на другую платформу.

Устойчивость

Программы, написанные на Natural, отличаются чрезвычайной надежностью. Развитый механизм обработки исключительных ситуаций позволяет программистам обнаруживать программные и системные. Благодаря архитектуре Natural не могут возникать эффекты "утечки" памяти (программы, запрашивающие памяти, но не освобождающие ее).

Эффективность

С самого начала Natural был реализован в архитектуре, которая сейчас называется "сервер приложений". Исходный код Natural компилируется в промежуточный код, который затем интерпретируется сервером приложений (Natural Runtime). Преимуществом является небольшой размер модуля по сравнению с естественным машинным кодом. Таким образом, программа требует небольшой объем памяти для выполнения. К тому же, небольшой размер модуля ведет к повышению скорости его загрузки. Загрузка программы и количество обращений к диску снижается также за счет использования буферного пула Natural, который позволяет хранить в памяти, часто используемые модули. В многопользовательских и многозадачных средах несколько пользователей совместно используют модули, расположенные в буферном пуле, что также уменьшает общее требуемое пространство памяти.

Это приобретает тем большее значение, что в организациях, где используется Natural, работают по 10 000 программ на Natural.

Операционная среда Natural

Natural является операционной средой, имеющей ядро, буферный пул, рабочие области и пр. Структура Natural, как операционной среды показана на рис. 6.3.

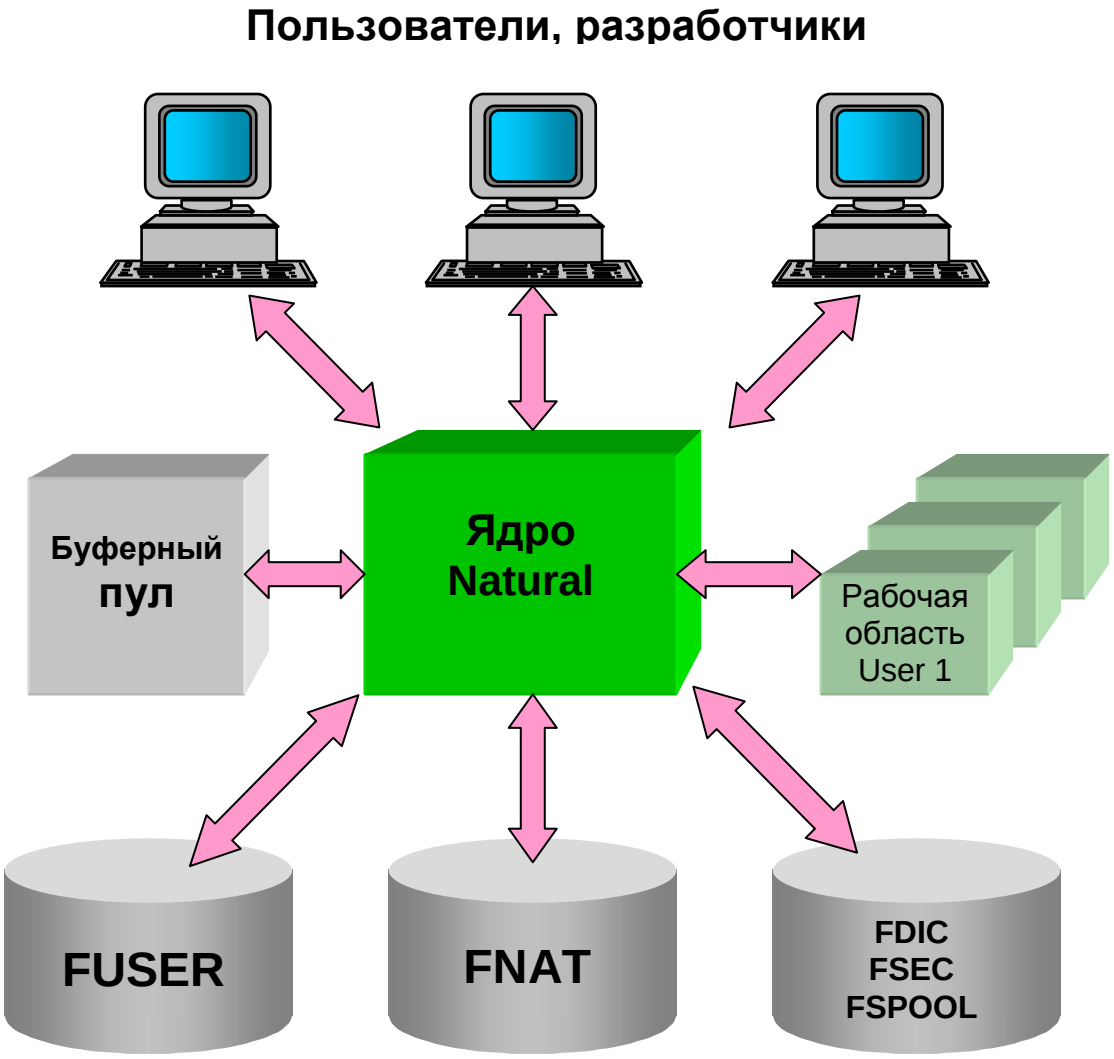


Рис. 6.3. Структура операционной среды Natural

Ядро

Системный администратор при инсталляции Natural производит сборку ядра Natural, включая необходимые системные модули. На платформе сервера ядро Natural запускается в составе телемониторов CICS, Complete и сервера разработки.

Рабочие области

Для каждого пользователя, который начинает работать с Natural и становится активным в оперативной памяти, Natural выделяет рабочая

область. В рабочей области содержатся: глобальные и локальные области данных; рабочие области для редакторов программ, шаблонов; управляющая информация.

Буферный пул

Объектные коды программ и шаблонов во время выполнения загружаются в буферный пул, если их там не было, где они становятся доступным другим пользователям для выполнения. При нехватке памяти в буферном пуле для загрузки необходимой программы, ядро Natural производит очистку памяти за счет удаления из буферного пула давно загруженных и не используемых объектных кодов.

Файл FUSER

Исходные тексты и объектные модули программ пользователей хранятся в библиотеках в файле FUSER.

FNAT

Объектные модули системных программ, утилит и других инструментарием хранятся в библиотеках в файле FNAT.

FDIC

Описание объектов, программ, модулей, систем, файлов, БД и т.д. хранятся в файле FDIC. Файл FDIC поддерживается посредством использования системы PREDICT.

Natural – в деталях

Основные компоненты Natural показаны на рис. 6.4.

Рис. 6.4. Основные компоненты Natural

Система помощи Natural

Система помощи (Help) (рис. 6.5.):

- Help доступен с любого экрана и активизируется нажатием клавиши F1, вводом символа “?” или команды “Help”.

- Содержит полную информацию о командах, редакторах, операторах, функциях, сообщений об ошибках.

- Позволяет, используя активную систему навигации найти нужную информацию.

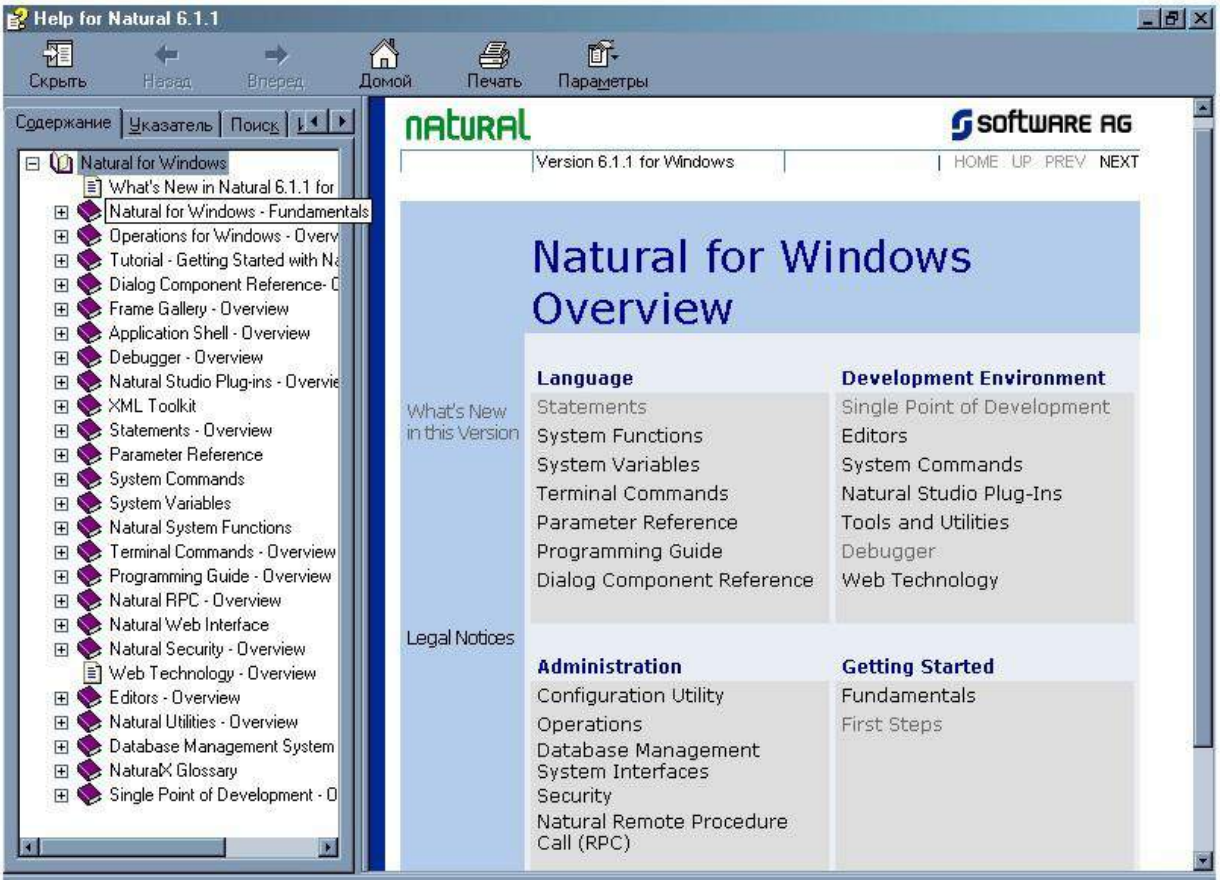


Рис. 6.5. Система помощи Natural

Возможности Системы помощи:

- Система помощи доступна с любого экрана и активизируется нажатием клавиши F1, вводом символа “?” в первом байте поля, вводом команды “Help” в командной строке, через функцию “help” панели инструментов.

- Нажатие клавиши F1 выдает справочную информацию о разделе или функции, с экрана которого была нажата клавиша.

- Система помощи содержит полную структурированию информацию о командах, редакторах, операторах, функциях, сообщений об ошибках.

- Система помощи позволяет, используя активную систему навигации быстро получить необходимую информацию.

- Для просмотра информации о возникшей ошибке необходимо ввести строке следующую информацию “? nnnn”, где nnnn – номер ошибки. Можно вводить без лидирующих нулей.

- На мэйнфрейм вызывается меню системы помощи, отмечая необходимые разделы можно дойти до нужной справочной информации.

- Система помощи может быть использована в обучении возможностям системы разработки Natural.

Документация Natural

Natural Programmer's Guide

Руководство Программиста

Эта документация относится ко всем платформам, на которых Natural может работать. Документация обеспечивает основную информацию о различных аспектах программирования в Natural.

Natural User Manual

Руководство Пользователя

Эта документация содержит информацию по использованию редакторов, системных команд и инструкций языка.

Руководство пользователя по системе Natural для Windows

Эта документация содержит информацию по использованию редакторов, системных команд и инструкций языка применительно для системы Windows.

Natural Reference Manual

Справочное Руководство

Эта документация содержит детальную информацию, необходимую для детального изучения синтаксиса операторов, параметров, системных переменных и терминальных команд.

Natural Statements Manual

Руководство по операторам

Эта документация содержит примеры использования для каждого оператора.

Natural Utilities Manual

Руководство по утилитам

Эта документация содержит информацию по использованию утилит обслуживания.

Natural Operations Manual

Руководство по настройке

Эта документация содержит информацию настройке параметров системы, настройки телемониторов, описания интерфейсов с базами данных.

Основные возможности редактора Natural Studio

Общий вид редактора Natural Studio показан на рис. 6.6.

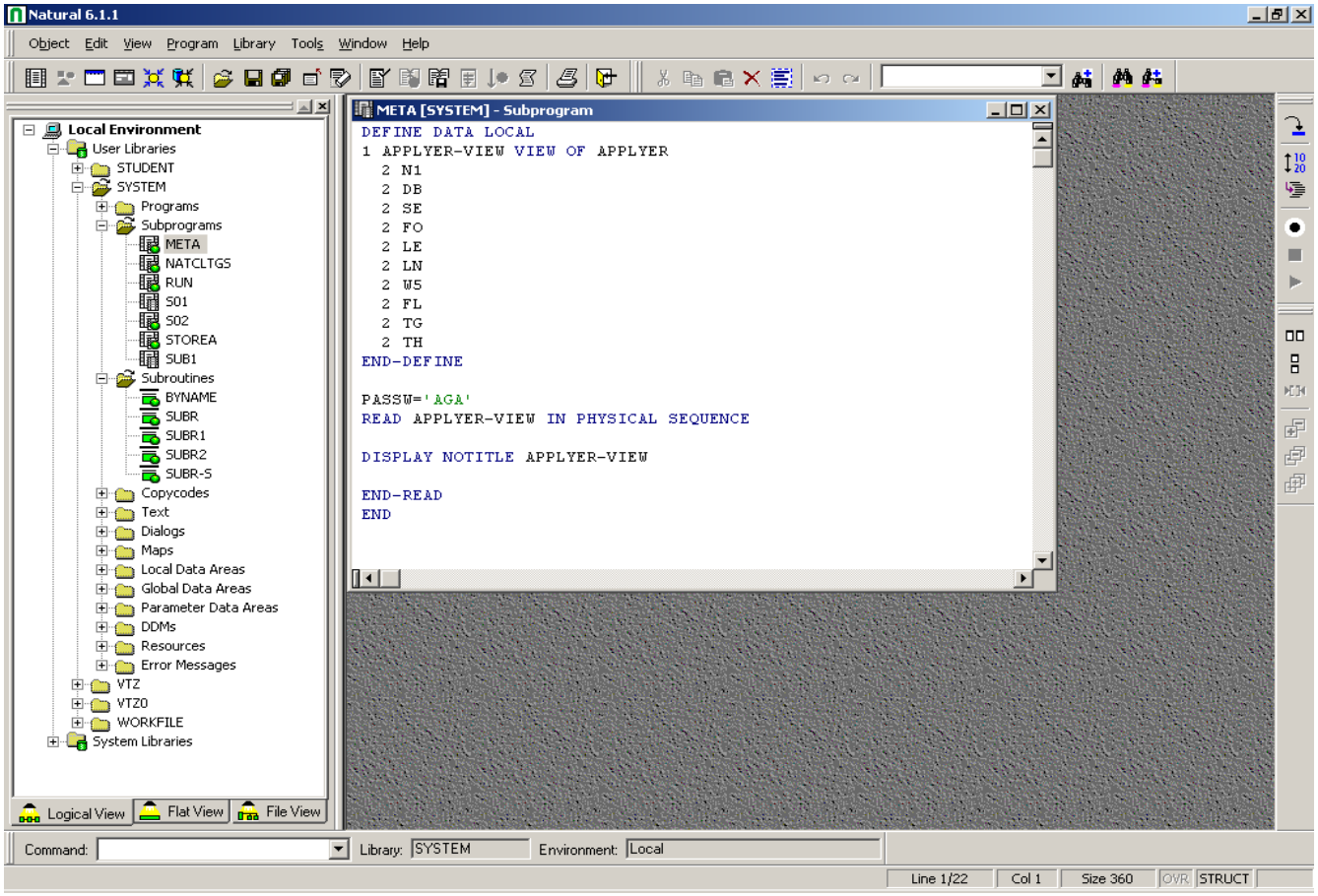


Рис. 6.6. Общий вид среды разработки Natural Studio

Структура файлов Natural отображается в виде дерева и может быть представлена в следующих режимах (рис. 6.7.):

1. *Режим логического просмотра (Logical View)*. Все файлы Natural (программы, подпрограммы и пр.) представляются проектировщику сгруппированными по логическим группам (папки ‘Programs’, ‘Subprograms’, ’Dialogs’ и пр.).

2. *Режим “плоского” просмотра (Flat View).* Все файлы Natural отображаются без группировки по логическим группам в алфавитном порядке.

3. *Режим файлового просмотра (File View).* Отображается физическая структура файлов Natural.

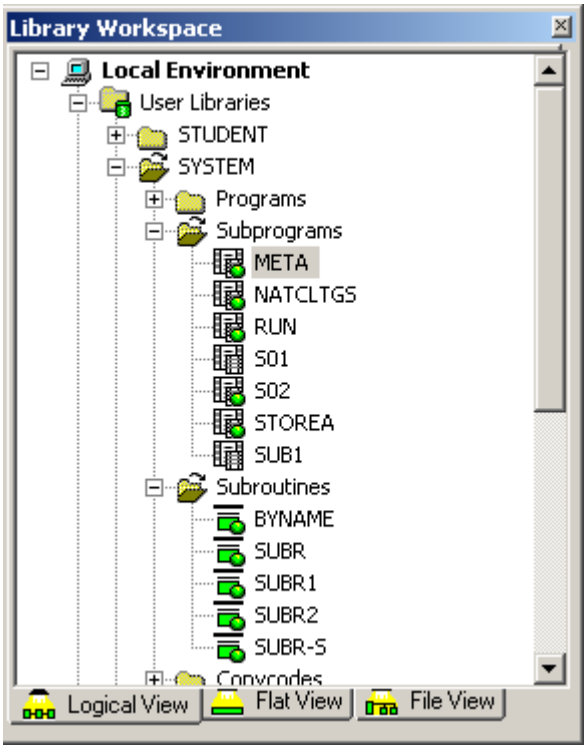


Рис. 6.7. Структура файлов в Natural Studio

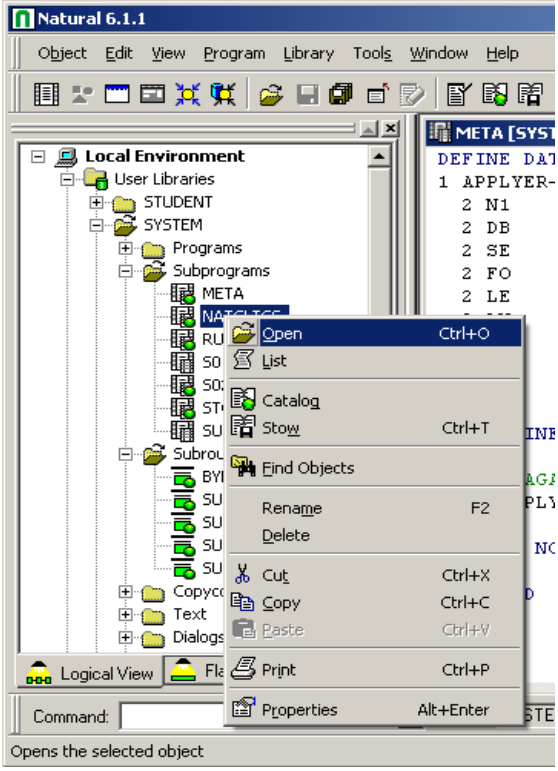


Рис. 6.8. Открытие файла в редакторе Natural Studio

Для редактирования файлов Natural необходимо выбрать соответствующий файл в каком-либо из режимов отображения и воспользоваться опцией контекстного меню ‘Open’ (рис. 6.8.). Можно также воспользоваться двойным нажатием левой кнопки мыши. При этом содержимое требуемого файла отобразится в рабочей области, автоматически открыв необходимый редактор для редактирования в соответствии с типом файла.

Рис. 6.9. Панель редактирования ‘Object’

Панель редактирования ‘Object’ (рис. 6.9.) служит для выполнения наиболее частых операций с файлами Natural:

Новая программа (New Program) – создание нового файла программного кода.

Новый класс (New Class) – создание нового класса Natural.

Новый диалог (New Dialog) – создание нового диалога Natural.

Новая карта (New Map) – создание нового файла карты данных.

Новая локальная область данных (New Local Data Area) – создание нового файла локальной области данных.

Новый DDM (New DDM) – создание нового файла DDM.

Открыть файл (Open) – открытие существующего файла Natural.

Сохранить (Save) – сохранение изменений в открытом файле.

Сохранить все (Save All) – сохранение изменений во всех открытых файлах.

Закрыть файл (Close) – закрытие активного файла Natural.

Отчистить файл (Clear) – создание нового файла; тип файла соответствует типу активного файла в момент выбора данной опции.

Проверка программного кода (Check) – проверка корректности синтаксиса, использованного в рамках создаваемого файла.

Добавить файл в каталог (Catalog) – каталогизирование (добавление активного файла в структуру каталога).

Наполнение файла (Stow) – сохранение и каталогизация файла.

Запуск файла (Run) – запуск программного кода к исполнению (опция доступна для модулей типа программа и диалог).

Исполнение файла (Execute) – запуск программного модуля к исполнению (опция доступна для модулей типа программа и диалог).

Просмотр программного кода файла (List) – просмотр исходного объекта Natural.

Печать (Print) – печать содержимого файла.

Выход из редактора (Exit) – закрытие редактора Natural с сохранением изменений во всех файлах.

7.2. Определение локальной области данных

Для осуществления операций запросов при помощи языка Natural первой процедурой должна быть процедура определения области данных. Под областью данных, в данном случае понимается конечный перечень переменных (с указанием их форматов и типов), которые будут использоваться в рамках данного программного средства (программы, диалога Natural).

Также в средствах редактора Natural предусматривается создание автономных глобальных, локальных областей и областей параметров, которые хранятся в виде отдельного модуля и могут использоваться несколькими приложениями Natural.

Для создания файла *локальной области* (*глобальной*, или *области параметров*) *данных* необходимо выбрать пункт меню *Новая локальная область данных/Новая глобальная области данных/Новая область параметров (New Local Data Area/New Global Data Area/New Parameter Data Area)* панель редактирования ‘Object’ (рис. 6.10.)

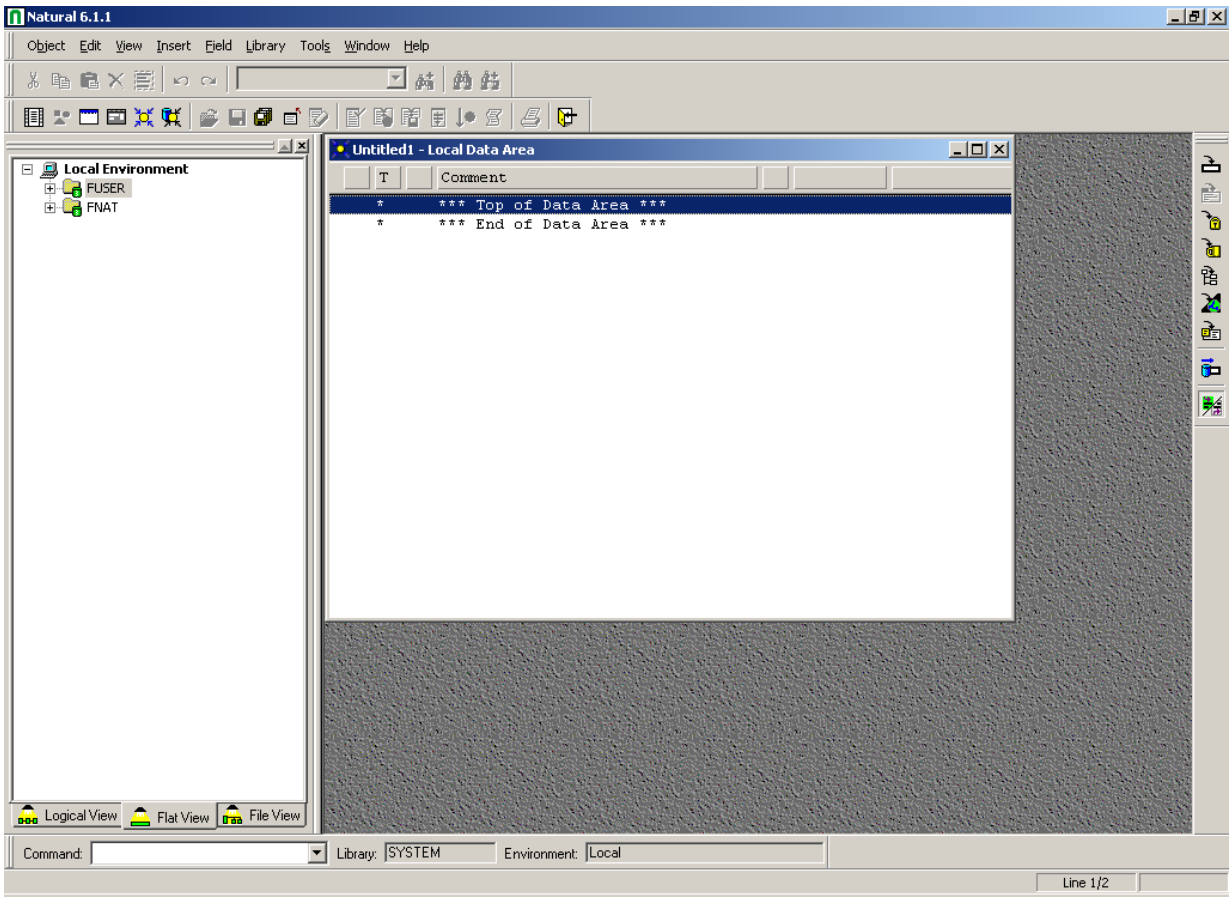


Рис. 6.10. Создание файла области данных

При этом в рабочей области редактора отображается окно для ввода и редактирования определяемых переменных (рис. 6.10.).

Редактор Области Данных

Редактор областей данных поддерживает:

- Глобальные области данных.
- Области, к которым могут обращаться несколько программных объектов, находящихся в одной библиотеке.
- Область параметров.
- Области, содержащие элементы данных, которые используются в качестве параметров в подпрограмме, внешней процедуре или диалоге.
- Локальные области данных.
- Области, содержащие элементы данных, которые используются, только в одном модуле Natural.

Область описания полей базы данных в пользовательской программе производится через логическое представление базы данных. *Создание области описания полей* базы данных всегда осуществляется через DDM, посредством выборки необходимых полей. В локальной области наименование полей, последовательность расположения полей могут быть изменены.

Структура заголовка столбца описания областей данных

Строка заголовка столбца (рис. 6.11.) содержит описание областей данных. Заголовок столбца изменяется, в зависимости от типа области, которая выбрана.

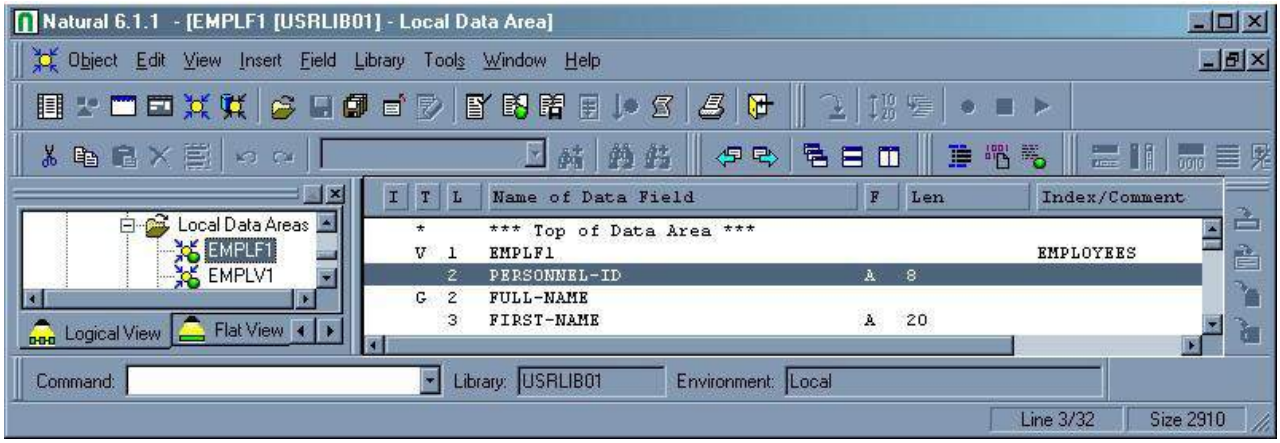


Рис. 6.11. Редактор области данных

Значения полей строк описания областей данных могут принимать следующие значения (табл. 6.1.):

Значения полей строк описания областей данных

Таблица 6.1.

Заголовок	Описание
I	Информационное поле:
	R - Поле является «результатом величины» параметров. (Только допущенные входные области данных параметра.)
	V – Поле является «величиной» параметра. (Только допущенные входные области данных параметра.)
	X - Больше информации доступна для поля. Маска редактирования, заголовок, или начальная величина определяется для поля.
T	Тип поля:
	B - Block
	C – Константа или счетчик переменной в View
	G – Групповое поле
	M – Множественное поле
	O - Объект
	P – Периодическая группа
	S - Структура
	U – Глобальный уникальный ID (GUID)
	V - View
	пробел – Простое поле
	* - Комментарий
L	Уровни поля (1 - 99).
Name of Data Field	Имя view, группы, периодической группы, множественного поля, константы, счетчика переменной, блока или простого поля
F	Формат. Любой формат, поддерживаемый в Natural.
Len	Длина. Значения длины нет для полей формата C, D, T и L. Можно определить в поле длины значение "DYNAMIC".
Index/Comment	Индекс поля - массива и комментарий области
Name of DDM	Имя DDM
Parent/Comment	Родитель блока и комментарий области

Представление данных в Natural

В программах Natural используются следующие типы данных:

– *Данные, определяемые пользователем*. Определенные пользователем данные применяются для хранения промежуточных результатов в программе или процедуре. Они объявляются в глобальных и локальных областях данных, так и могут объявляться непосредственно в программе. Для этих данных объявление формата и длины обязательны.

– *Данные, определяемые системой Natural*. Это системные переменные, значения которых автоматически определяются Natural. Имена системных переменных начинаются с “*”. Объявление формата и длины системным переменным не требуется, они определены в Natural.

– *Данные, определяющие поля базы данных*. Поля базы данных определяются в локальных областях данных на основе модуля описания данных DDM (data definition module). Объявление формата и длины полям базы данных не требуется, они определены в модуле DDM.

Правила присвоения имен:

1. Имя переменной может иметь длину от 1 до 32 символов.
2. Первым символом имени должен быть один из следующих символов:
   - ПРОПИСНАЯ БУКВА.
   - # чтоб отличить от зарезервированных слов (рекомендуется).
   - + переменных в глобальной области данных.
   - & для динамического формирования исходного текста программы.

Имя переменной может иметь длину от 1 до 32 символов. В некоторых случаях можно использовать имена длиной более 32 символов (например, в комплексных приложениях для удобства чтения программ); однако, важны только первые 32 символа, которые должны быть уникальны, остальные символы система Natural игнорирует.

Имя переменной пользователя не должно совпадать с зарезервированным словом системы Natural.

Имя переменной пользователя не должно совпадать с именем поля базы данных в пределах одной программы Natural, так как это может привести к погрешностям ссылки.

Для того чтобы сослаться на предыдущий оператор Natural, используется метка оператора или номер строки исходного текста с данным оператором. Эту возможность используют для изменения системных ссылок, принятых по умолчанию (описывается для каждого оператора), или для документирования.

Если не определена ссылка на оператор, то действует следующее правило: по умолчанию берутся поля базы данных, прочитанные в активном внутреннем цикле обработки (FIND, READ или HISTOGRAM). Если поле не читалось в активном цикле, то оно берется из предыдущего оператора GET (в режиме отчета также из оператора FIND FIRST или FIND UNIQUE), в котором было прочитано.

Описание формата и длины определяемых переменных показаны в таблице 6.2.

Описание формата и длины переменных

Таблица 6.2.

Формат		**Определяемая длина (кол-во цифр)**	**Внутренняя длина (в байтах)**
A	Алфавитно-цифровой	**1 - 1073741824 (1GB)**	1 - 1073741824
B	Двоичный	**1 - 1073741824 (1GB)**	1 - 1073741824
C	Управление атрибутом	-	2
D	Дата	-	4
F	С плавающей точкой	**4** или **8**	4 или 8
I	Целый	**1**, **2** или **4**	1, 2 или 4
L	Логический	-	1
N	Числовой распакованный	**1 - 29**	1 - 29
P	Числовой упакованный	**1 - 29**	1 - 15
T	Время	-	7

Команды обслуживания объектов

Перечень и описание команд обслуживания объектов показан на рис. 6.12. и в табл. 6.3.

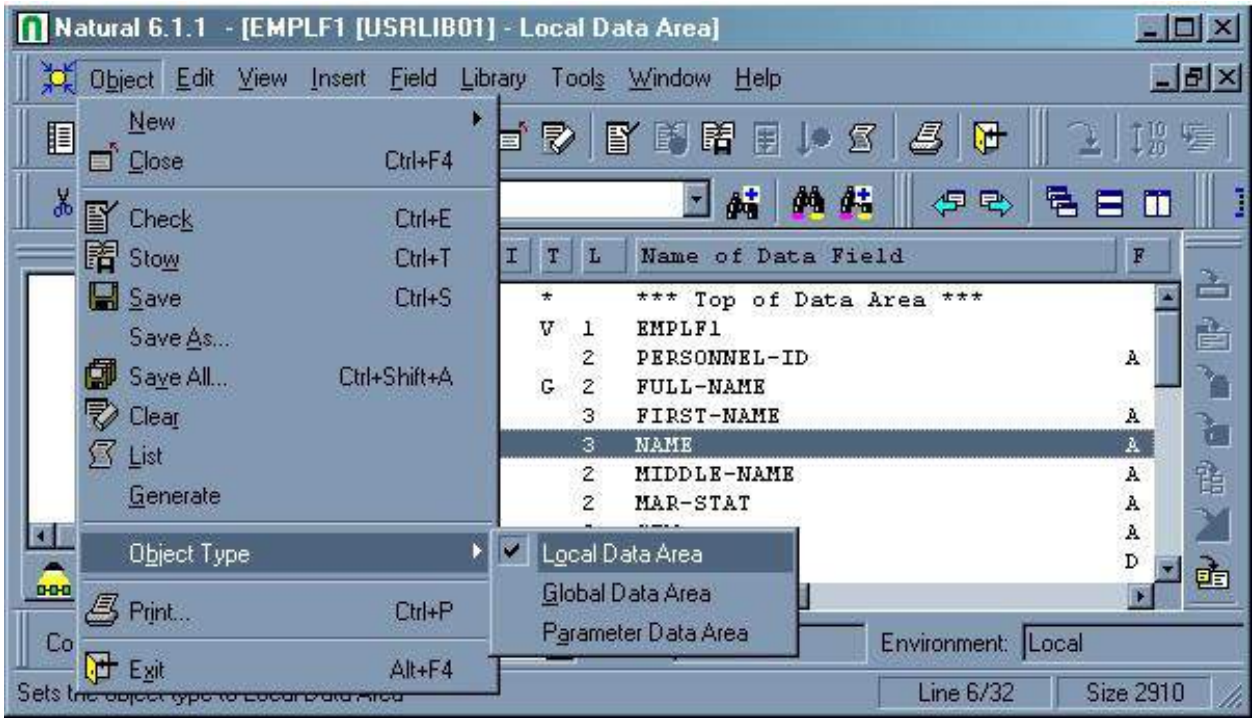


Рис. 6.12. Команды обслуживания объектов

Команды по работе с объектами

Таблица 6.3.

Опция	Команды по работе с объектом
New	Создать новый объект: программы, области, и т.д.
Close	Закрыть рабочую область
Check	Проверить содержимого редактора на синтаксические ошибки
Save	Сохранить содержимое редактора в исходом коде
Save As …	Сохранить содержимое редактора в другой библиотеке или с другим именем и типом
Save All …	Сохранить все объекты в исходном коде
Clear	Очистить рабочую область
List	Просмотреть исходный код объекта
Generate	Сгенерировать copy cod из содержимого редактор
Object Type =>	Выбрать и изменить тип объекта: Local, Global, Parameter
Print	Распечатать содержимое на принтере
Exit	Выйти из редактора и сеанса Natural

Команды редактирования объекта

Перечень и описание команд редактирования объектов показан на рис. 6.13. и в табл. 6.4.

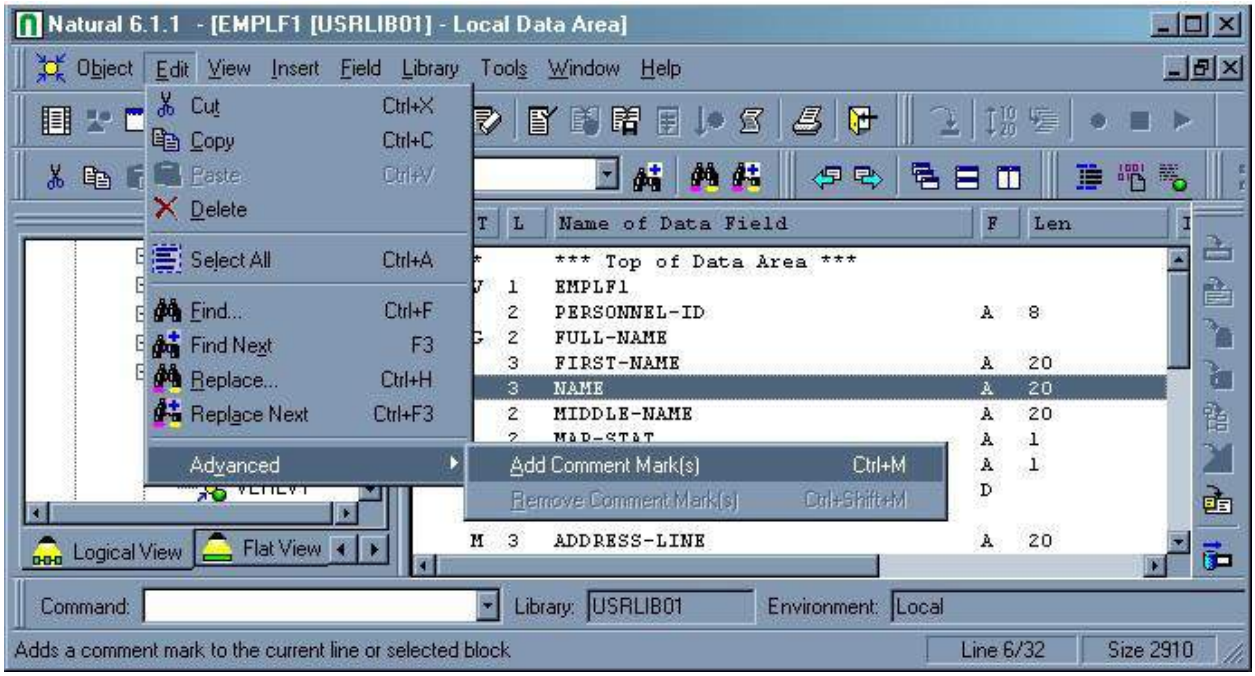


Рис. 6.13. Команды редактирования объекта

Команды редактирования объекта

Таблица 6.4.

Обозначение	**Команды редактирования объекта**
Cut	Вырезать
Copy	Скопировать
Paste	Вставить
Delete	Удалить
Select All	Выбрать все
Advanced	Установить или снять метку комментария с поля

Команды вставки данных

Перечень и описание команд вставки данных показан на рис. 6.14. и в табл. 6.5.

Рис. 6.14. Команды вставки данных

Команды вставки данных

Таблица 6.5.

Обозначение	**Команды вставки данных**
Data field	Вставка нового поля после выбранной курсором строки
Block …	Вставка строки блока
Constant …	Вставка строки константы
Handle …	Вставка элементов типа Object или Dialog
Structure …	Вставка строки верхнего уровня структуры
Global Unique ID	Функция только возможна с LDA и GDA
Comment …	Вставка строки комментариев
Import … =>	Импортировать определение полей из других объектов

Посредством опции *импортирования данных (Import…)* возможна вставка полей из других объектов Natural в структуру создаваемой области данных (рис. 6.15.)

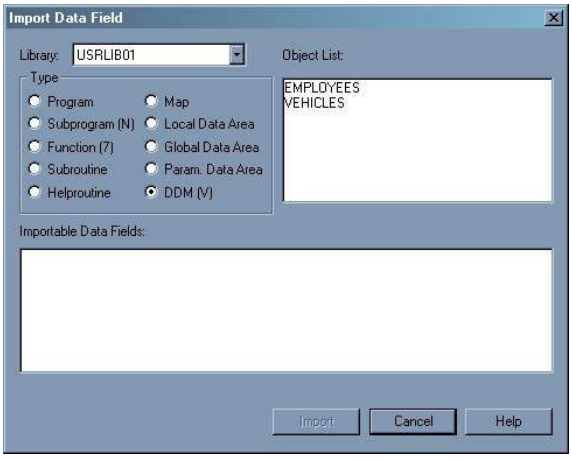


Рис. 6.15. Импортирование данных

Команды изменения поля области данных

Перечень и описание команд изменения поля области данных показан на рис. 6.16. и в табл. 6.6.

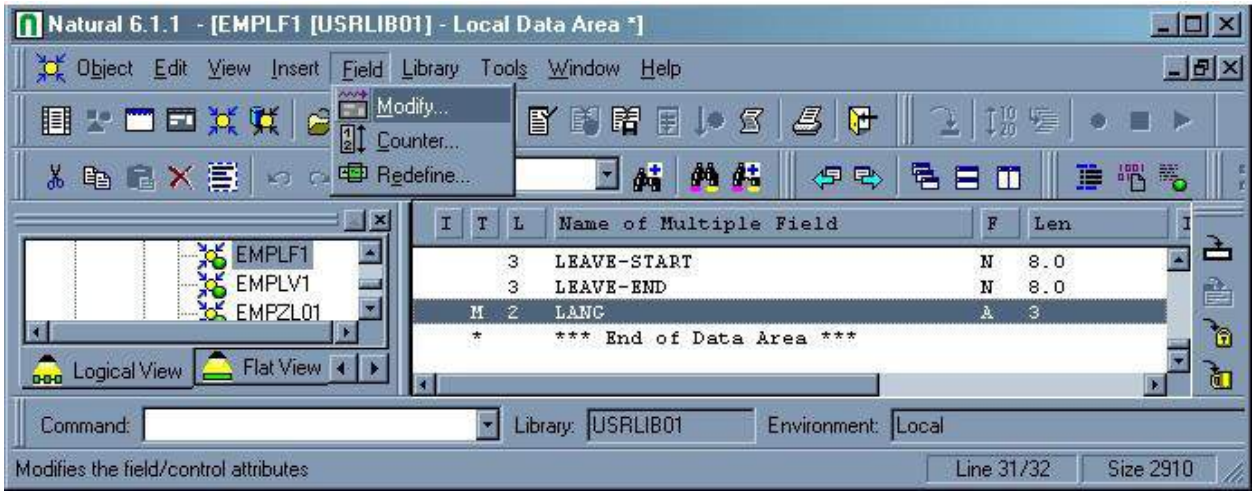


Рис. 6.16. Команды изменения поля области данных

Команды изменения поля области данных

Таблица 6.6.

Обозначение	**Команды изменения поля области данных**
Modify …	Изменить поле (рис. 6.17.)
Counter …	Определить число полей для периодической группы (рис. 6.18.)
Redefine …	Переопределить поле (рис. 6.19.)

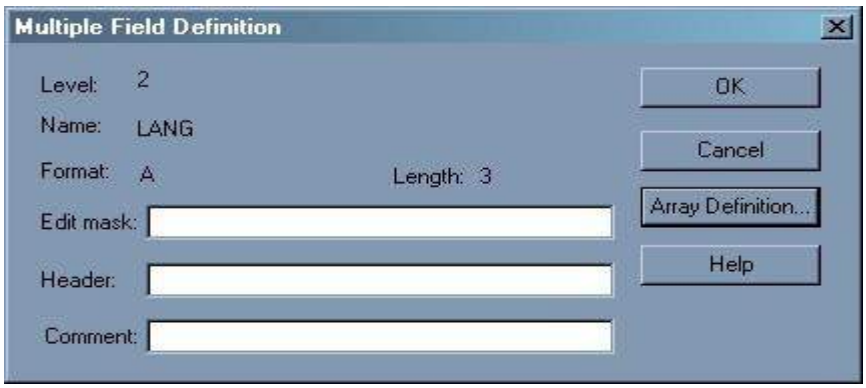


Рис. 6.17. Изменение поля

Рис. 6.18. Определение числа полей для периодической группы

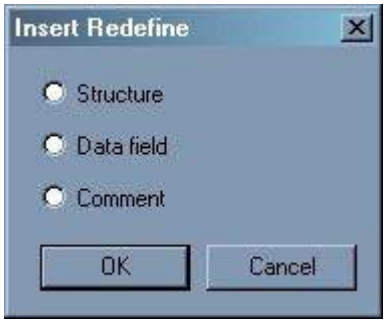


Рис. 6.19. Переопределение поля

Настройка параметров редактора данных

Служба настройки параметров редактора данных вызывается из меню Tools > Options > Data Area Editor (рис. 6.20.)

Рис. 6.20. Параметры редактора данных

Настройки параметров редактора данных включают следующие опции:

Insert before - включить область перед выбранной областью.

Insert after - включить область после выбранной области.

Status bar - строка статуса редактора области данных. Показ строки статуса:

(1) - количество байтов использовались областью данных в исходной области;

(2) - позицию выбранной строки;

(3) - общее число строк в области данных.

Редактор области данных может оперировать максимуме 9999 строк.

Column header - заголовок столбца на верхе определения области данных.

Fixed fonts only - используйте этот выбор, чтобы определить, что только фиксированные шрифты должны быть выбираемы в диалоговое меню "Шрифт".

Font - используйте этот выбор, чтобы выбрать тип шрифта для заголовка столбцов редактора области данных.

6.3. Создание DDM файла БД

Редактор DDM (Data Definition Modules) – модуль определения данных, который позволяет создать и поддерживать модуль определения данным любого тип файла базы данных.

В рамках редактора DDM для модулей из одной библиотеки доступны модули из той-же библиотеки и из всех "общедоступных", перечисленных в параметре STEPLIB, в параметрах среды исполнения Natural.

DDM (Модуль определения данных) описывает таблицу базы данных, это может быть описание как таблицы в СУБД Adabas, так и в других СУБД. DDM используются, чтобы описать любой тип файла базы данных, и не ограничивается файлами базы данных Adabas. Данный обеспечивает связь проектируемых приложений с БД.

Экран редактора DDM

При открытии файлов DDM (опция *Open* панели управления ‘Object’) окно редактора будет отображено для каждой области DDM (рис. 6.23.).

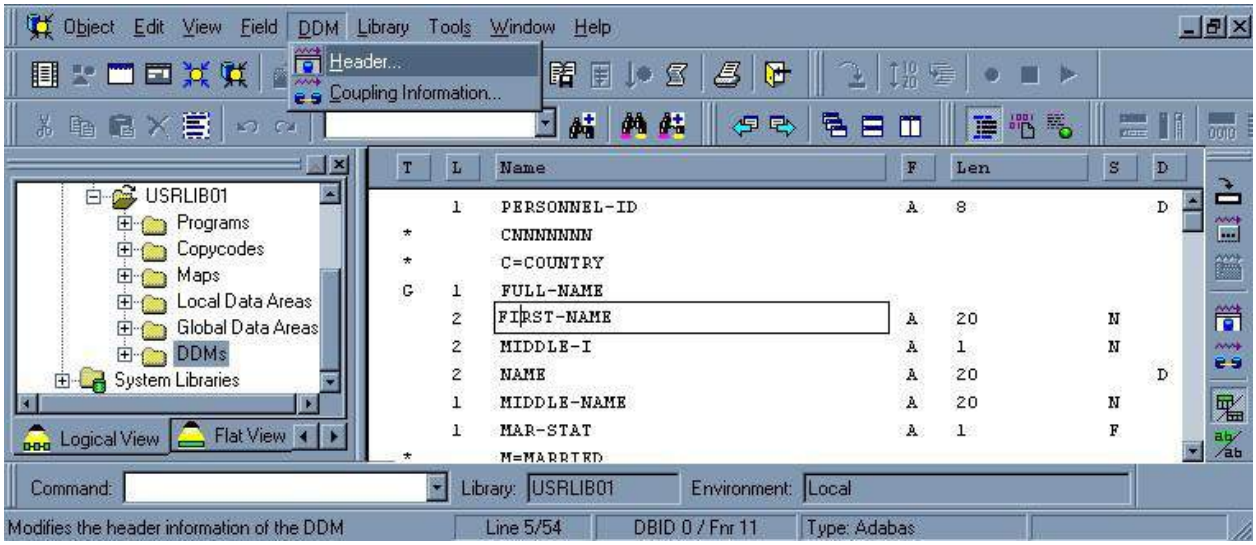


Рис. 6.23. Окно редактора DDM

Для создания нового файла DDM необходимо воспользоваться опцией *New DDM* панели управления ‘Object’. При этом в последующем окне (рис. 6.24.) необходимо указать:

- тип базы данных, к файлу которой создается DDM;
- идентификационный номер БД;
- идентификационный номер файла БД.

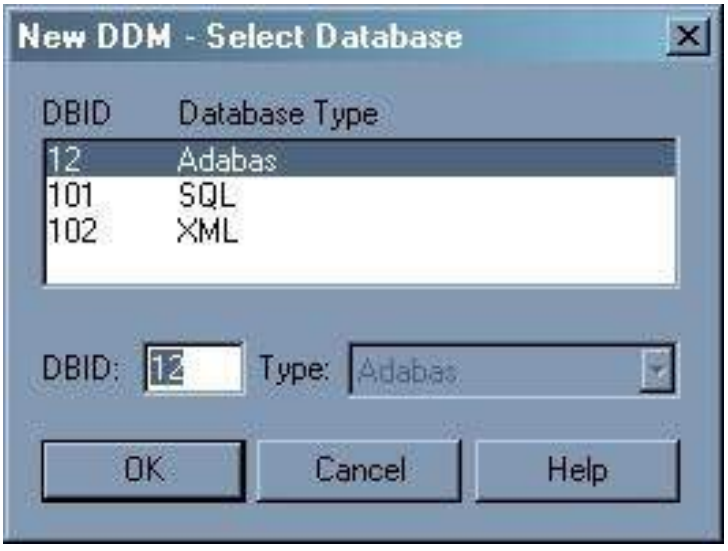


Рис. 6.24. Создание нового DDM

В зависимости от типа БД, используемой в качестве базы для создания DDM, могут использоваться следующие типы DDM (таблица 6.8.).

Типы DDM

Таблица 6.8.

Тип DDM (сокращение)	**Тип БД**
A	Adabas
V	VSAM
2	DB2
D	DL/I
P	Entire System Server
C	Command Processor
S	Super Natural
X	XML
E	Entire DB Engine

Заголовок областей DDM

После создания DDM можно приступить к форматированию *заголовков областей DDM* (рис. 6.25.)

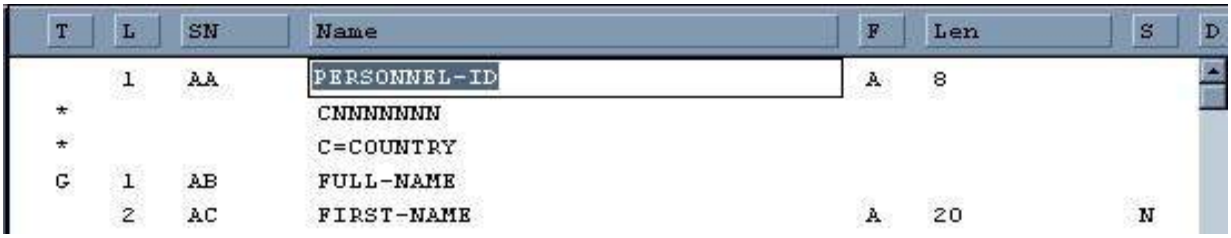


Рис. 6.25. Заголовки областей DDM

Элементы строки заголовка областей DDM показаны в таблице 6.9.

Элементы строки заголовка

Таблица 6.9.

Элементы	Описание
T	Тип поля: G - групповое поле M - множественное поле P - периодическое поле * - комментария Пробел - простое поле
L	Уровни поля (1 - 7)
SN	Для файлов Adabas двух - символьное имя поля
Name	Внешнее имя области 3 - 32 символов используемое в Natural программах
F	Формат, поддерживаемый в Natural: F, B, D, F, I, L, N, P, T
Len	Длина Стандартной области. Эта длина может быть переопределена в Natural программе.
S	Подавление величины (только для файлов Adabas) N - определяется с подавлением M - не подавляется F - фиксированная длина Пробел - не указывается подавление области
D	Опция описания Дескриптора: D - дескриптор S - супердескриптор или субдескриптор H - гипердескриптор N - не дескриптор P - фонетический дескриптор Пробел - простое не дескрипторное поле

Команды по работе с объектом

Команды по работе с объектом выполняют функции:

- создания;
- проверки;
- сохранения;
- каталогизации;
- печати объекта.

Команды по работе с объектом доступны пользователю через панель управления ‘Object’ или через опцию ‘Object’ главного контекстного меню (рис. 6.26.)

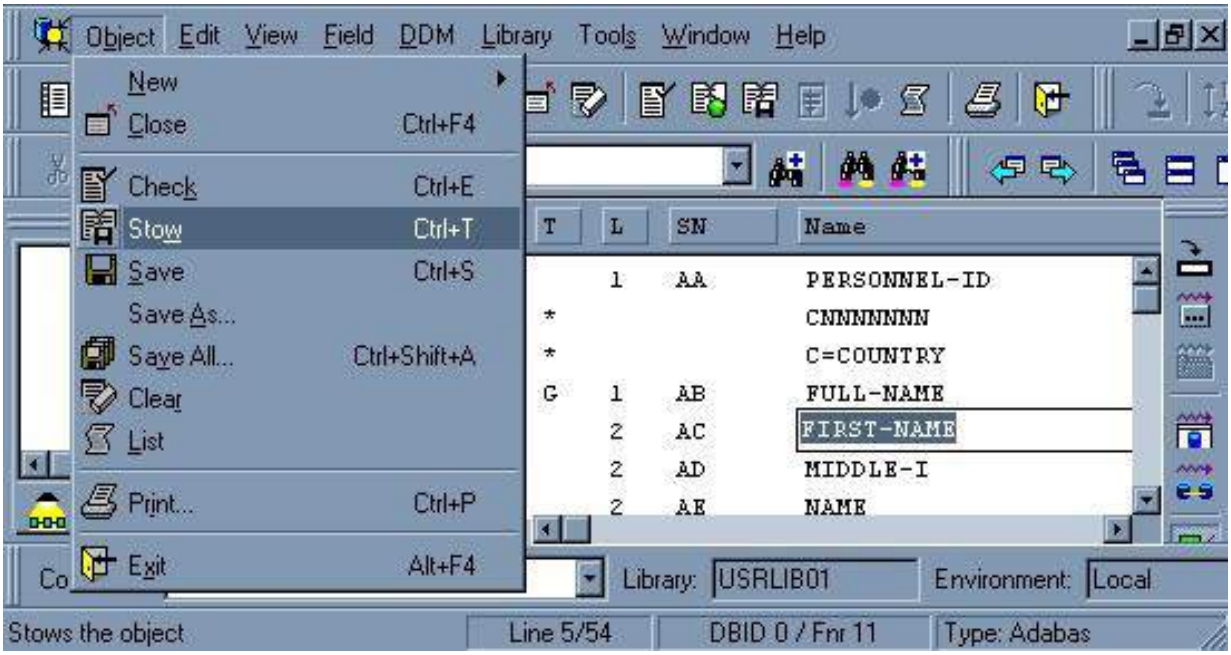


Рис. 6.26. Команды по работе с объектом

Перечень и описание команд представлены в таблице 6.10.

Команды по работе с объектом

Таблица 6.10.

Обозначение	Описание
New	Создать новый объект: программы, области, и т.д.
Close	Закрыть рабочую область
Check	Проверить содержимого редактора на ошибки
Stow	Сохранить и каталогизировать
Save	Сохранить содержимое редактора в исходом коде
Save As …	Сохранить содержимое редактора в другой библиотеке
Save All …	Сохранить все объекты в исходном коде
Clear	Очистить рабочую область
List	Просмотреть исходный код объекта
Print	Распечатать содержимое на принтере
Exit	Выйти из редактора и сеанса Natural

Команды редактирования

Команды редактирования выполняют функции:

- выбора;
- копирования;
- вставки;

- удаления объекта в библиотеке.

Команды редактирования доступны пользователю через опцию ‘Edit’ главного контекстного меню (рис. 6.27.)

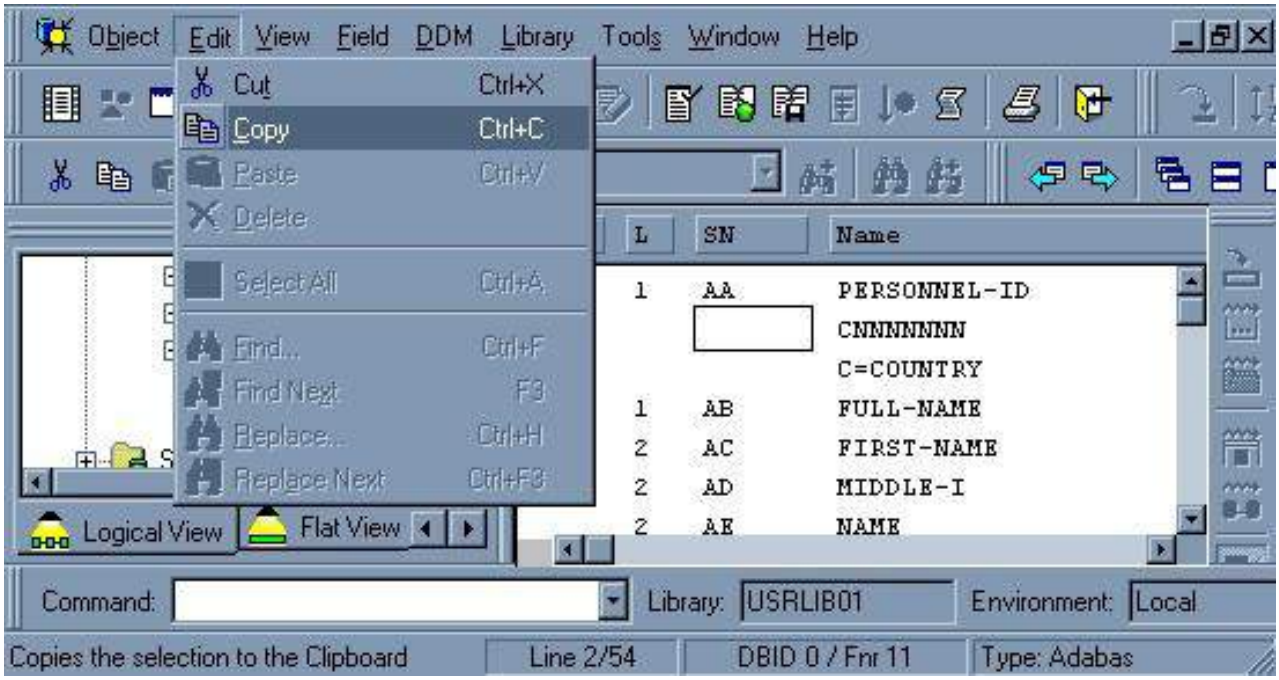


Рис. 6.27. Команды редактирования

Перечень и описание команд представлены в таблице 6.11.

Команды редактирования

Таблица 6.11.

Обозначение	Описание
Cut	Вырезать
Copy	Скопировать
Paste	Вставить
Delete	Удалить
Select All	Выбрать все

Команды изменения поля области данных

Команды изменения поля области данных выполняют функции:

- определения поля;
- вставки поля;
- создание ссылок суб и супердескрипторов.

Команды изменения поля области данных доступны пользователю через опцию ‘Field’ главного контекстного меню (рис. 6.28.)

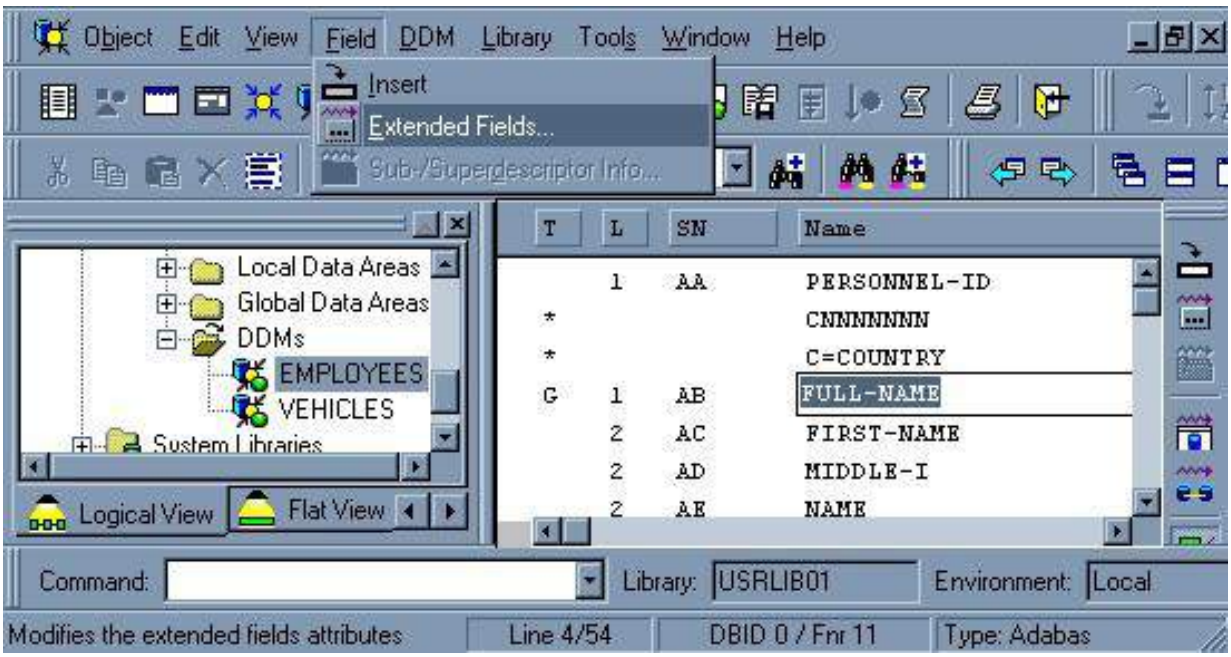


Рис. 6.28. Команды изменения поля области данных

Перечень и описание команд представлены в таблице 6.12.

Команды изменения поля области данных

Таблица 6.12.

Обозначение	**Описание**
Insert	Определение поля и присвоение ему характеристик
Extended Field	Команда вставить новое поле
Sub/Superdescriptor	Создание ссылок суб / супердескрипторов

При создании ссылок суб и супердескрипторов следует указывать начальные и конечные элементы посредством *окна редактирования дескрипторов* (рис. 6.29.)

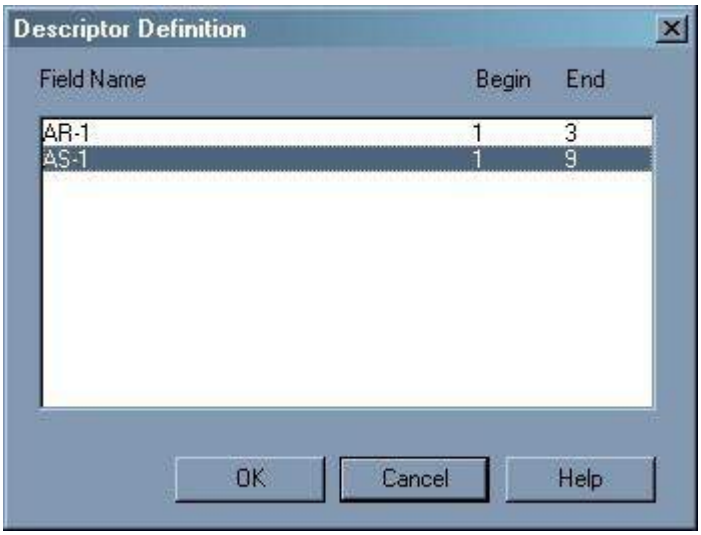


Рис. 6.29. Редактирование дескрипторов

Команды управления параметрами DDM

Команды управления параметрами DDM доступны пользователю через опцию ‘DDM’ главного контекстного меню (рис. 6.30.)

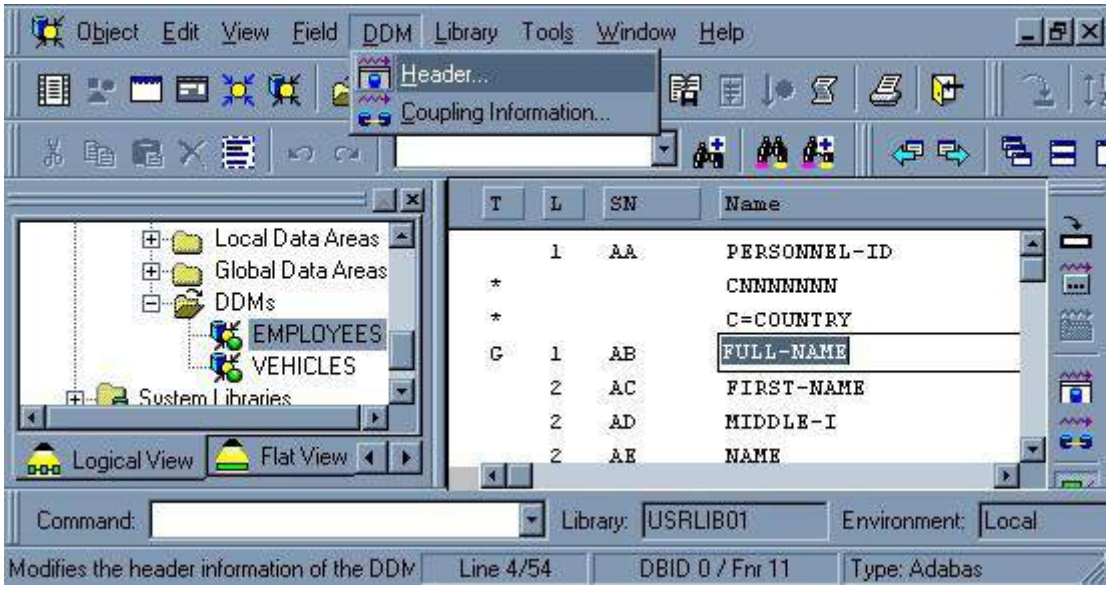


Рис. 6.30. Команды управления параметрами DDM

Перечень и описание команд представлены в таблице 6.13.

Команды управления параметрами DDM

Таблица 6.13.

Обозначение	Описание
Header	Просмотр и корректировка заголовка DDM
Coupling Information	Связывание файлов

Редактирование заголовка DDM осуществляется посредством последующего окна (рис. 6.31.) после выбора соответствующей опции (‘Header’).

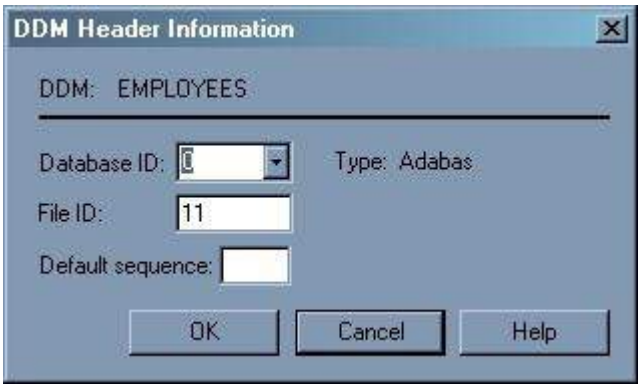


Рис. 6.31. Редактирование заголовка DDM

При редактировании заголовка DDM производится изменение следующих параметров:

DDM - имя текущего DDM;

DBID - ID Базы Данных от которой создан DDM;

FNR - номер файла базы данных от которой DDM создан;

Type - тип базы данных от которой DDM создан;

Default Sequence - поле, которая управляет логическим последовательным чтением файла, когда никакая область не определяется в операторе READ Natural программы.

Настройка параметров редактора DDM

Служба настройки параметров редактора DDM вызывается из меню Tools > Options > DDM Editor (рис. 6.32.)

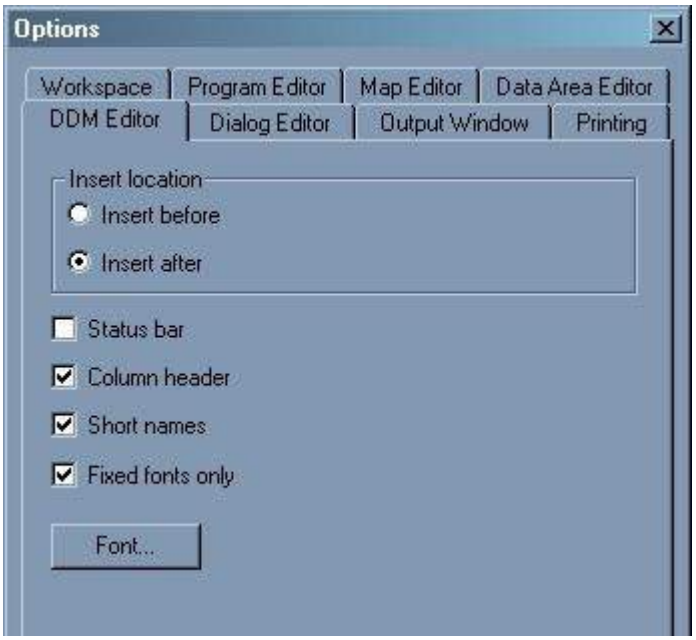


Рис. 6.32. Параметры редактора DDM

Настройки параметров редактора данных включают следующие опции:

Insert before - включить области перед выбранным полем;

Insert after - включить области после выбранного поля;

Status bar - включения/отключение отображения строки статуса редактора;

Column header - заголовок столбца модуля определения данных;

Short names - включения/отключение отображения коротких (2-х байтовых) имен полей в модуле определения данных;

Fixed fonts only - используйте этот выбор, чтобы определить, что только фиксированные шрифты должны быть выбираемы в диалоговом меню "Шрифт";

Font - используйте этот выбор, чтобы выбрать тип шрифта для заголовка столбцов модуля определения данных.

6.4. Редактирование программ Natural

Редактор программ поддерживает следующие типы объектов:

- программы (programs);
- классы (classes);
- подпрограммы (subprograms);
- функции (function);
- подпрограммы (subroutines);
- копикоды (copy code);
- помощи (help routines);
- тексты (text).

Редактор текстов программ используется, для создания и редактирования объектов Natural, таких как программы, подпрограммы, классы, копикоды, помощи и текстовые элементы. При использовании многочисленных окон редактирования, возможно копировать или перемещать содержание из одного окна редактирования в другое. Заголовок окна каждого объекта редактирования отображает имя и тип объекта. Если объект новый и еще не сохранен, этому объекту автоматически дается название "Untitled".

Меню редактора программ на Windows

Меню редактора программ содержит следующие функции и команды (рис. 6.33., таблица 6.14.)

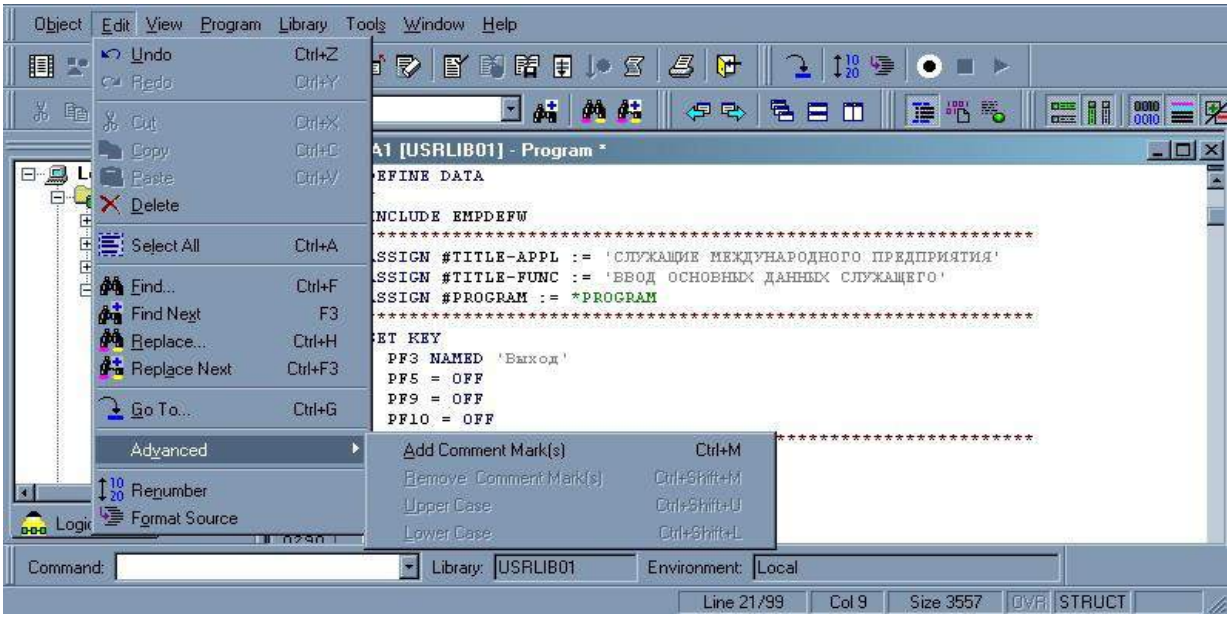


Рис. 6.33. Меню редактора программ

Меню редактора программ

Таблица 6.14.

Object	**Edit**	**Program**	**Library**
New => Close Check Save Save As … Save All … Clear Run **Object Type =>** Print Exit	Undo Redo Cut Copy Paste Delete Select All Go to … **Advanced =>** Renumber Format Source **Programs** **Subprograms** **Function** **Subroutines** **Copy code** **Help routines** **Text**	Open List **Import =>** **Add Mark** **Remove Mark** **Upper Case** **Lower Case**	New Cat All Find Object **Library: _____** **Program** **Subprogram** **Function** **Subroutine** **Help routine** **Map** **Local data area** **Global data area** **Parameter data** **DDM (V)**

Описание действий, выполняемых командами, представлено в таблице 6.15.

Команды редактора программ

Таблица 6.15.

Обозначение	**Описание**
Object	*Команды по работе с объектом*
New	Создать новый объект: программы, области, и т.д.
Close	Закрыть рабочую область
Check	Проверить содержимого редактора на синтаксические ошибки
Save	Сохранить содержимое редактора в исходом коде
Save As …	Сохранить содержимое редактора в другой библиотеке или с другим именем и типом
Save All …	Сохранить все объекты в исходном коде
Clear	Очистить рабочую область
Run	Выполнить программу из содержимого редактор
Object Type =>	Выбрать и изменить тип объекта: programs, subprograms, subroutines, classes, copy code, help routines and text
Print	Распечатать содержимое на принтере
Exit	Выйти из редактора и сеанса Natural
Edit	*Стандартные команды Windows*
Undo	Отменить последнее изменение
Redo	Вернуть отмену последнего изменения
Cut	Вырезать
Copy	Скопировать
Paste	Вставить
Delete	Удалить
Select All	Выбрать все
Advanced =>	Установить или снять метку комментария, преобразовать строку в строчные буквы или в заглавные буквы
Renumber	Перенумеровать строки программы в редакторе
Format Source	Форматировать строки программ с относительным смещением
Program	
Open	Открыть программу
List	Просмотреть листинг программы
Import … =>	Импортировать определение полей из других объектов
Library	*Команды по работе с библиотекой*
New	Создать новую библиотеку
Cat All	Выполнить каталогизацию объектов библиотеки
Find Object	Найти объект в библиотеке по имени, по внутреннему тексту

Программы в Natural пишутся в *структурном* или *отчетном режиме* программирования.

Установку режимов или их переключение можно производить несколькими способами:

- Можно произвести установку, используя Tools/Session Parameters/Compiler Options значение параметра SM=On/Off, что означает установку структурного режима или отчетного режима программирования.
- Переключать режим программирования можно кнопкой, которая устанавливается на панель меню из окна Tools/Customize/Toolbars/Compiler Options при отметке знаком «галочка».
- Можно задавать в строке прямых команд значения GLOBALS SM=ON или GLOBALS SM=OFF, что тоже производит переключение режимов.

Возможности редактора программ

Возможности редакторы программ (рис. 6.34.) могут быть сгруппированы по следующим группам:

- Модификация текста программ.
- Синтаксическая помощь.
- Разбиение окна редактора.
- Редактирование/ просмотр ссылочных Natural объектов.
- Запись/переигровка нажатий клавиш.
- Просмотр исходной программы в расширяемом/сжимаемом виде.

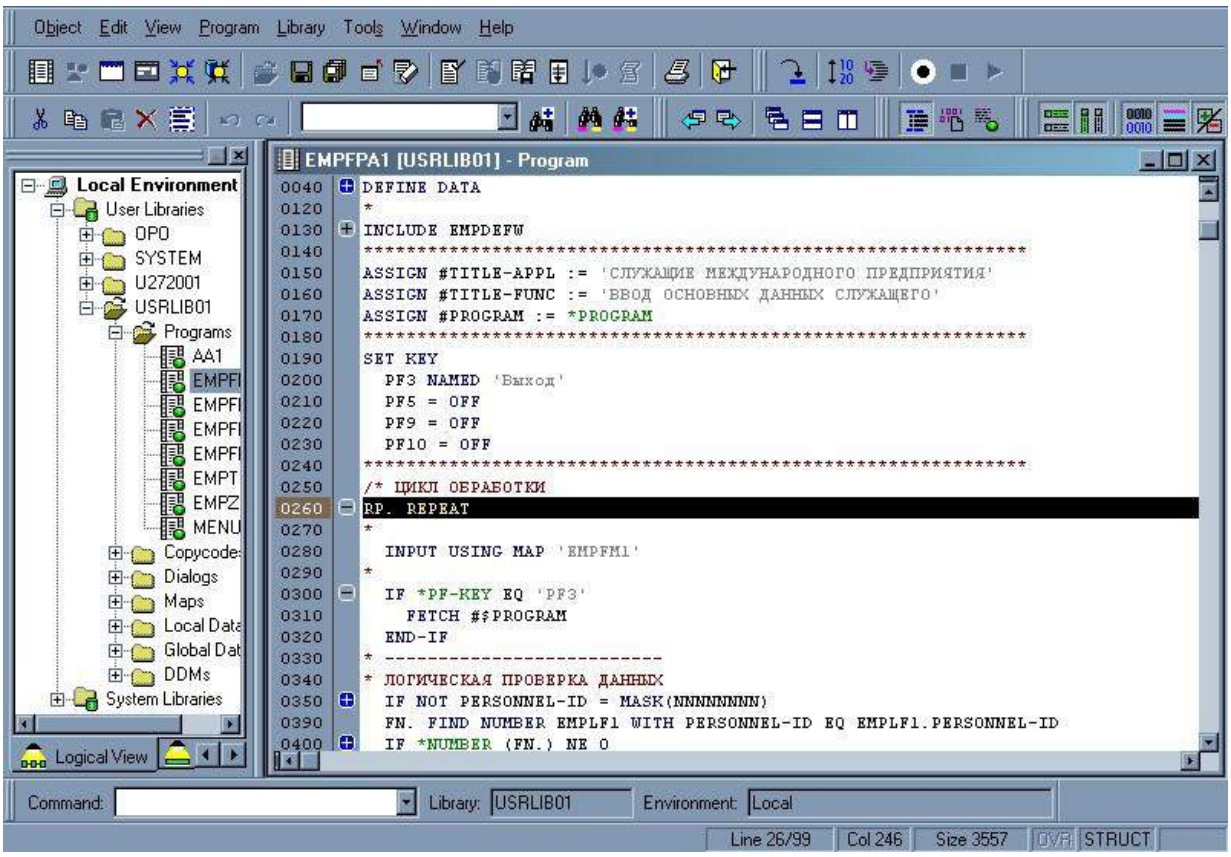


Рис. 6.34. Возможности редактора программ

Команды модификации текста программ:

- Поиск.
- Копирование.
- Вырезание.
- Вставка.
- Удаление.
- Восстановление и переделывание изменений.
- Перенумерация строк программы.

Также предусмотрены дополнительные команды модификации текста (меню Edit > Advanced)

- Add Comment Mark(s) – добавить как строку комментариев.
- Remove Comment Mark(s) – снять метку комментария.
- Upper Case – превратить в заглавные символы.
- Lower Case – превратить в строчные символы.

Редактирование/ просмотр ссылочных Natural объектов

Пока Вы работаете в редакторе текстов программ, Вы можете открыть или указать другие объекты Natural, которые указываются в программном коде. Если, например, Вы редактируете программу, которая вызывает подпрограмму, можно открыть подпрограмму, связать ее с программой, и возвратиться в программу. Чтобы отредактировать ссылочный объект Natural в программе нужно выбрать команду из меню Program > Open Object.

Чтобы просмотреть ссылочный объект Natural в программе нужно выбрать команду из меню Program > Open List.

Запись/воспроизведение нажатий клавиш

Вы используете функцию записи/воспроизведения, чтобы записать и воспроизвсти последовательности нажатия клавиш, которые Вы хотите повторить в редакторе текстов программ. Это подобно магнитофону с функциями запись, остановка и воспроизводить.

Чтобы *записать* последовательность нажатия клавиши, следует произвести несколько последовательных операций:

- Установка курсора в редакторе в позиции, с которой необходимо начать запись.
- Из меню Tools, выбирается опция Start Recording (одновременное нажатие клавиш Ctrl+Shift+R).
- Сообщение последовательности клавиш, которую необходимо записать.
- Будет записан только ввод данных с клавиатуры. Перемещения Мыши и операции будут проигнорированы.

Остановка записи:

- Из меню Tools, выбирается Stop Recording (одновременное нажатие клавиш Ctrl+Shift+S).

Переигровка записанной последовательности нажатия клавиш:

- Установка курсора в редакторе в позиции, с которой необходимо начать переигрывание записи.
- Из меню Tools выбирается Replay Recording (одновременное нажатие клавиш Ctrl+Shift+P).

Синтаксическая Помощь

Для вызова помощи выберите мышью одно из перечисленных данных и нажмите F1.

В пределах редактора текстов программ Natural, доступна *контекстно-зависимая помощь* для следующих синтаксических Natural элементов:

- Системные переменные.
- Системных функций.
- Операторы.
- Параметры.

Разделение Окна Редактора

Вы можете разделить окно редактора текстов программ вертикально или горизонтально, чтобы рассмотреть и модифицировать две части объекта одновременно. Изменения отображаются в обоих частях окна.

Чтобы разделить окно редактора на две области нужно выбрать команду из меню View > Vertical split (Horizontal split).

Возврат в один экран View > Unsplit.

Переключение между экранами делается выбором окна мышкой или клавишей F6.

Просмотр исходного кода программы в расширяемом/сжимаемом виде

Исходная программа не отображается по умолчанию в *расширяемом/раскладном формате*. Вы должны определить при выборах редактора текстов программ, что исходная программа должна отображаться в расширяемом/раскладном формате. Эта характеристика прилагается только в

исходной программе Natural, объектов кроме DDM, которые сохраняются в структурном способе:

- образы с минус (-) указывать начало сжимаемого блока программы;
- образы с плюс (+) указывать начало расширяемого блока программы.

Для просмотра исходной программы в расширяемом/сжимаемом виде, необходимо выбрать установку из меню Tools > Options > "Program Editor" > "Expand/Collapse". Выполнить команду, расширяемую/сжатие можно через соответствующие клавиши (таблица 6.16.):

Команды расширения/сжатия

Таблица 6.16.

Клавиши	**Функция**	**Результат**
Ctrl + "B"	Начало блока	Расширить/сжать соответствующий блок.
Enter	Сжать блок	Включает новую строку ниже сжатого блока.
Shift + Enter	Сжать блок	Расширяет блок и включает новую строку
Ctrl + "+"	Сжать блок	Расширьте текущий блок.
Ctrl + Alt + "+"	Сжать блок	Расширьте текущие и все последующие блоки.
Ctrl + "-"	В пределах блока	Сжать текущий блок.
Ctrl + Alt + "-"	В пределах блока	Сжать текущие и все последующие блоки.

Свертывание/расширение программных структур производится из меню: View > Collapse Block / View > Expand Block.

Установка опций редактора программ

Опции редактора программ вызываются из меню Tools > Options > Program Editor (рис. 6.35.).

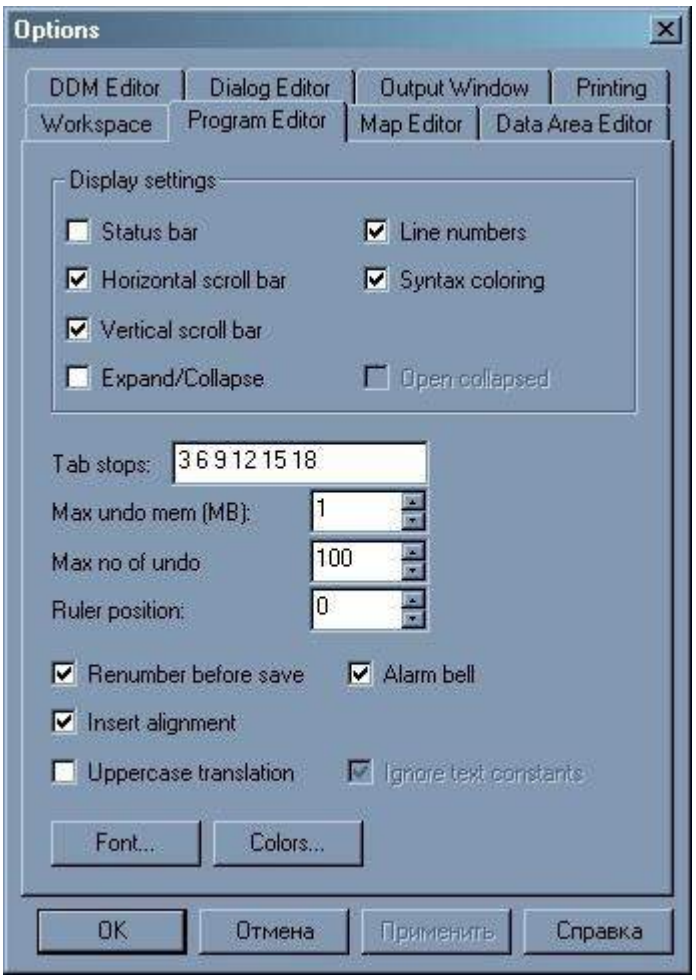


Рис. 6.35. Опции редактора программ

Назначение параметров опций редактора программ:

Status Bar - отображает строку статуса на верхе окна редактора.

Line Numbers - показывает колонку числа в редакторе текстов программ.

Syntax coloring - отображение цветного- кодирования для программных элементов.

Vertical scroll bar - отображает вертикальную зону прокрутки для окна редактора.

Horizontal scroll bar - отображает горизонтальную зону прокрутки для окна редактора.

Tabs - определение номера столбца для остановов табулятора в редакторе текстов программ.

Expand/Collapse - допустимые логические блоки программы в объектных листингах, которые должны переключаться между сжатой формой и

расширенной формой. Эта опция влияет на отображение программных объектов написанных в структурном режиме.

Open collapsed - ввод разрешен, если выделен переключатель "Expand/ Collapse". Первоначально отобразите логические блоки программы в сжатой форме в листинге программы.

Max. Number of actions - определение количества текстовых операций, которые могут быть восстановлены при использовании функции отмена/переделка. Нуль значит максимальный предел подчиненный только в определенном размере максимальной памяти.

Max. Memory size - определение буферного пространства доступного для хранения текстовых операций сделанного в редакторе текстов программ. Нуль значит доступное максимальное пространство.

Alarm - использование звукового сигнала в случае, когда нажата неправильная клавиша или ключевая комбинация.

Insert Alignment - выравнивание курсора с первым символом предшествующей строки текста.

Renumber Before Save - перенумерация строк в программе перед каждым сохранением.

Uppercase Translation - включается перевод в верхний регистр строк исходной программы перед сохранением.

Ignore Text Constants - игнорирует перевод текста, заключенного в кавычки, в верхний регистр.

Установка цвета синтаксиса текста

Для установки цвета (рис. 6.36.) следует выбрать из меню Tools > Options > Program Editor > Colors.

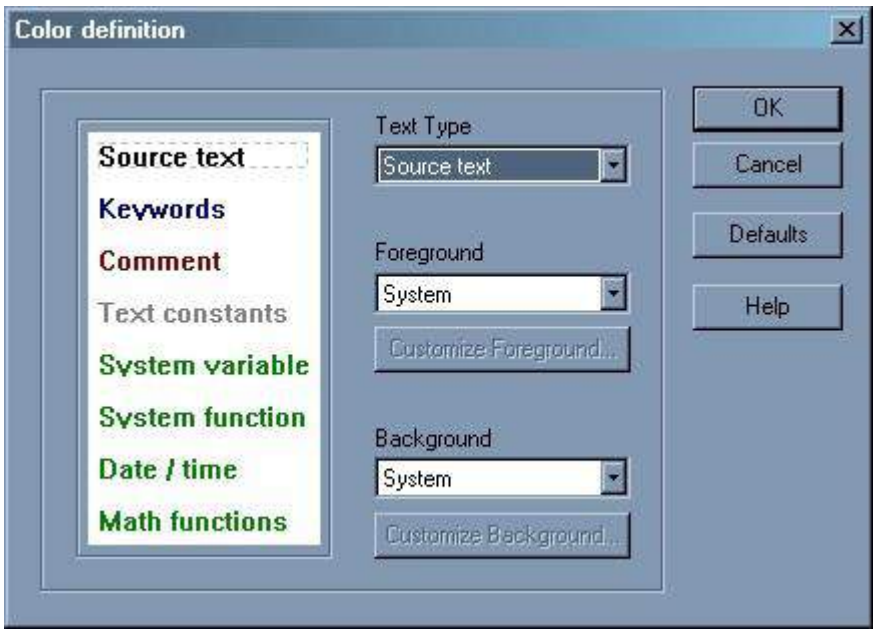


Рис. 6.36. Установка цвета синтаксиса текста

Раскраска в редакторе текстов программ используется, чтобы выделить различные элементы синтаксиса для лучшей удобочитаемости.

Синтаксические цвета, устанавливаемые по умолчанию в тексте программ Natural следующие:

- Синий - ключевые слова Natural.
- Красные - комментарии.
- Зеленый - текстовые константы, системные функции, системные переменные.
- Черные - определяемые пользователем переменные и остальные синтаксические элементы.

Можно определить и назначить собственные цвета.

Цвета переднего и заднего планов:

- Foreground - передний план.
- Background - задний план (фон).

6.5. Экранные формы ввода

Редактор экранных шаблонов и выходных форм

Возможности редактора экранных шаблонов и форм:

- позволяет создавать понятный, дружественный интерфейс пользователя для работы пользователей;

- позволяет заметно уменьшить трудозатраты разработчиков приложений на создание форматированных экранов ввода и вывода;
- позволяет использовать широкий диапазон элементов;
- обладает эффективными командами редактирования.

После того, как шаблон создан и сохранен в библиотеке, его можно использовать в программах оператором INPUT USING MAP.

Особенности построения шаблона экранов

При создании шаблонов проектируемого приложения необходимо воспользоваться методом создания *типовой структуры шаблона*. Типовая структура шаблона позволит стандартизовать вид представления данных, значения клавиш, переключателей и окошек выбора данных.

При создании других шаблонов необходимо задать в параметре Layout имя типового шаблона (например, Layout MAP01). Если задать в поле признака Dynamic=Y, то все изменения эталонной формы шаблона будут автоматически проведены в его использованных копиях; если оставить в поле Dynamic=N, то копии будут получены статически, и изменения эталонной формы не отражаются в его копиях.

Для управления отображением значений полей на экране, можно применить следующие переменные типа Control:

- AD – атрибуты отображения;
- CD – атрибуты цвета;
- CV – атрибуты переменного значения.

Значения атрибутов CV и AD показаны в таблице 6.19.:

Значения атрибутов CV и AD

Таблица 6.19.

Код	Атрибуты
	Атрибуты Представления
B	Blinking - мигание
C	Cursive/Italic – курсив рукописный
D	Default - нормальный
I	Intensified – повышенной яркости
N	Non-display – не отображаемый
U	Underlined - подчеркивание
V	Reverse video – обратное изображение
	Атрибуты выравнивания значения
L	выравнивание слева
R	выравнивание справа
Z	0 выровнен справа
	Атрибуты ввода/вывода
A	поле ввода, не защищено
M	поле выводимое, изменяемое
O	поле выводимое, защищено
P	временно защищено
	Преобразование символов в строчные или заглавные буквы
T	преобразование в заглавные буквы
W	преобразование в строчные буквы
	Атрибуты цвета
BL	Blue – голубой
GR	Green – зеленый
NE	Neutral – нейтральный
PI	Pink - розовый
RE	Red - красный
TU	Turquoise – бирюзовый
YE	Yellow – желтый

Создание и заполнение окошка (Selection Box) выбора значений

Для создания окошка выбора значений необходимо выполнить следующую последовательность действий:

- Выбрать в описании поля (Field definition) опцию Items.
- Заполняя поле Item необходимым значением (реквизитом), выбрать опцию Add Constant (или Add Variable) соответственно.

- Выбрав опцию Import Item данные можно импортировать из программ, локальных областей, других шаблонов и т.д.

Команды по работе с объектом

Перечень и описание команд по работе с объектом показан на рис. 6.38. и в табл. 6.20.

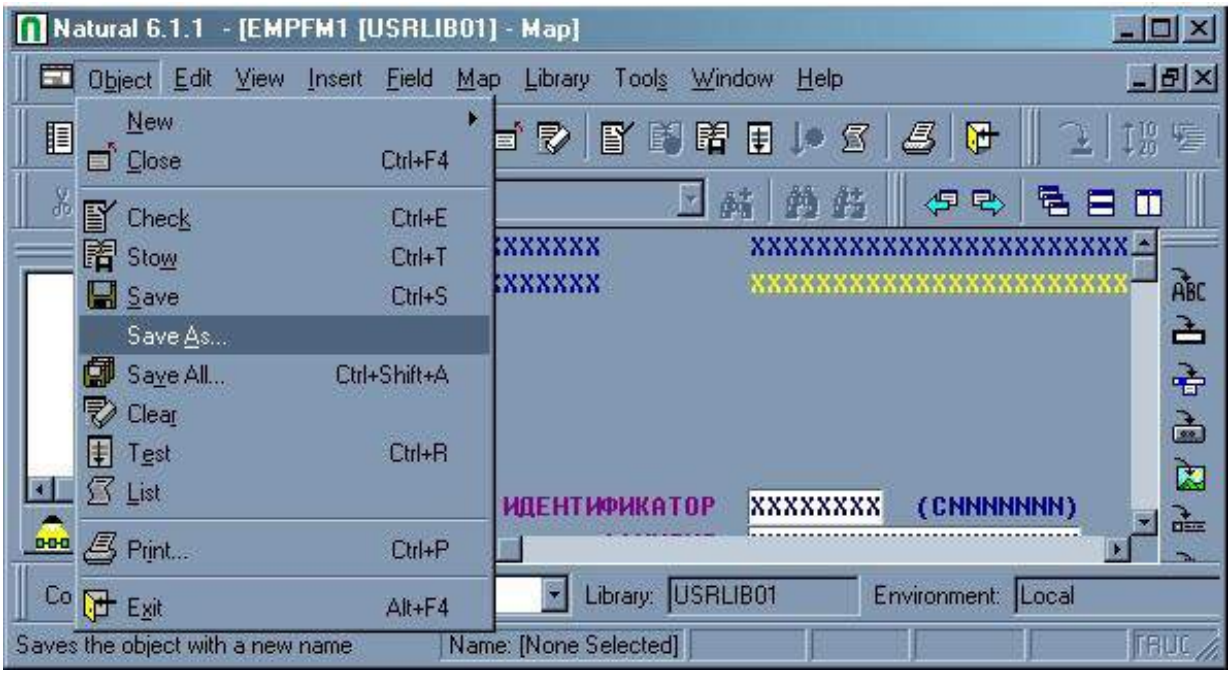


Рис. 6.38. Команды по работе с объектом

Команды по работе с объектом

Таблица 6.20.

Обозначение	Описание
New	Создать новый объект: программы, области, и т.д.
Close	Закрыть рабочую область
Check	Проверить содержимого редактора на синтаксические ошибки
Save	Сохранить содержимое редактора исходным кодом
Save As …	Сохранить содержимое редактора в другой библиотеке или с другим именем
Save All …	Сохранить все объекты в исходном коде
Clear	Очистить рабочую область
Test	Просмотреть создаваемый шаблон, каким он будет во время выполнения
List	Просмотреть программный код шаблона
Print	Распечатать содержимое на принтере
Exit	Выйти из редактора и сеанса Natural

Команды редактирования объекта

Перечень и описание команд редактирования объекта показан на рис. 6.39. и в табл. 6.21.

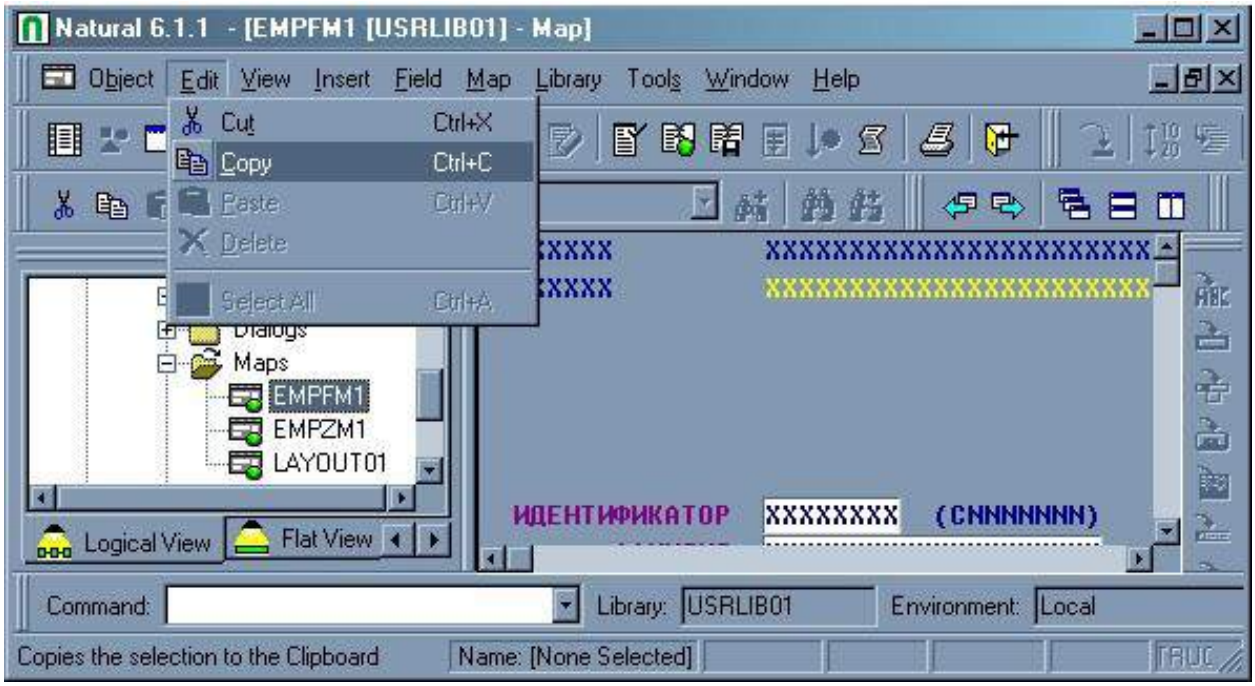


Рис. 6.39. Команды редактирования объекта

Команды редактирования объекта

Таблица 6.21.

Обозначение	Описание
Cut	Вырезать
Copy	Скопировать
Paste	Вставить
Delete	Удалить
Select All	Выбрать все

Команды вставки данных

Перечень и описание команд вставки данных показан на рис. 6.40. и в табл. 6.22.

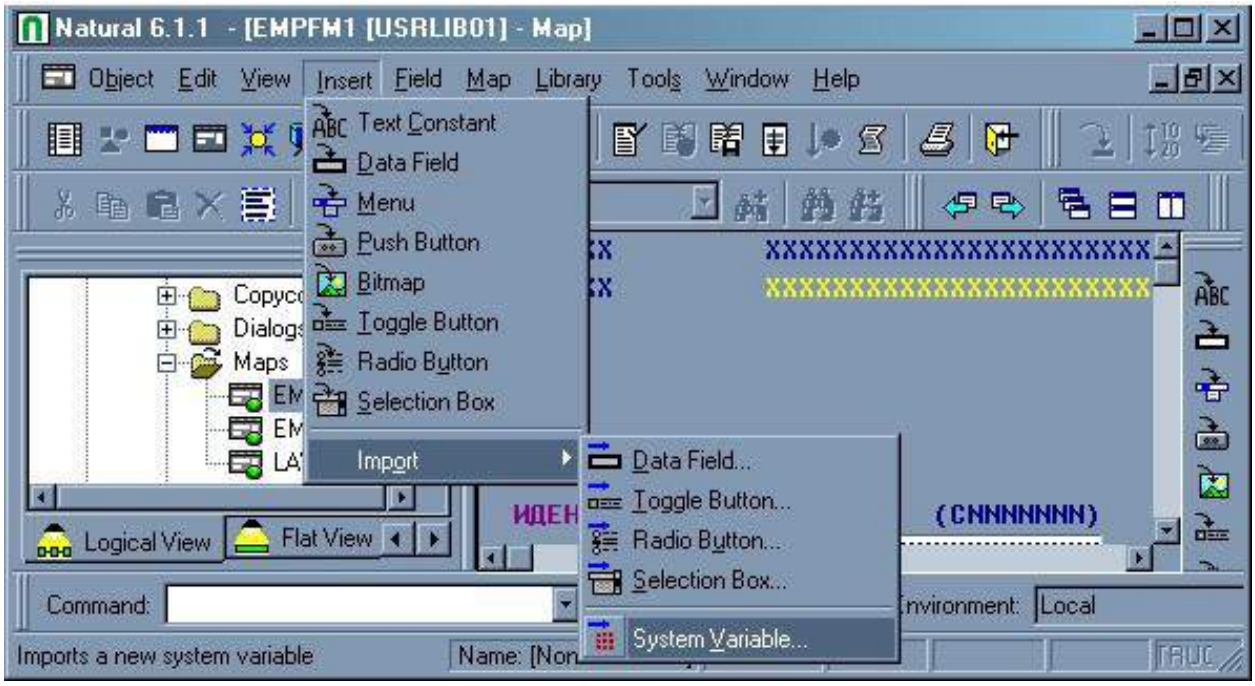


Рис. 6.40. Команды вставки данных

Команды вставки данных

Таблица 6.22.

Обозначение	Описание
Text Constant	Вставка нового текста в экранную форму
Data Field	Вставка поля данных форматов используемых в Natural
Menu	Вставка экранной формы меню с заголовком
Push Button	Вставка кнопок
Bitmap	Вставка картинок
Toggle Button	Вставка кнопки переключателя
Radio Button	Вставка кнопки включателя
Selection Box	Вставка окошка для выбора значений
Import … =>	Импортировать данные из других объектов и системную переменную

При импортировании данных из других объектов, в последующем окне (рис. 6.41.) необходимо указать имена соответствующих полей.

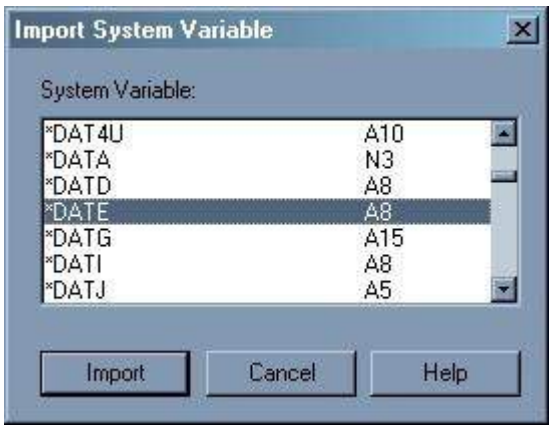


Рис. 6.41. Импортирование данных

Команды изменения поля области данных

Перечень и описание команд изменения поля области данных показан на рис. 6.42. и в табл. 6.23.

Рис. 6.42. Команды изменения поля области данных

Команды изменения поля области данных

Таблица 6.23.

Обозначение	Описание
Definition	Определение поля и присвоение ему характеристик
Color	Присвоения цвета и характеристик отображения
Rules	Создание и присвоение полю правил проверки значений
Alignment =>	Выравнивание поля по разным направлениям

Определение поля производится при помощи ввода параметров в соответствующем окне (рис. 6.43.)

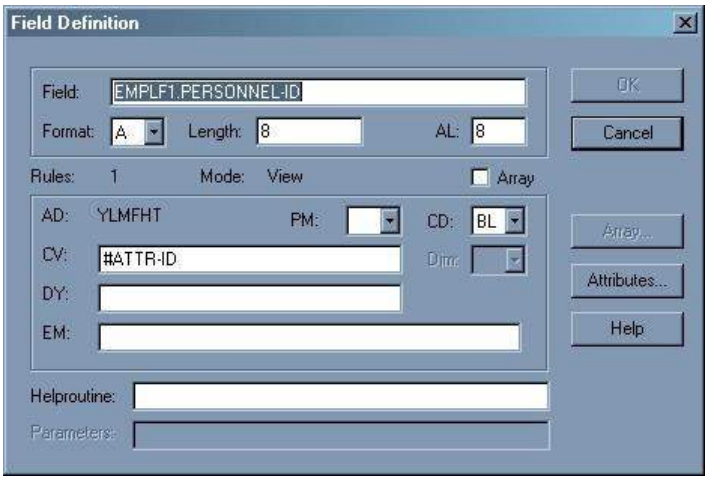


Рис. 6.43. Определение поля и присвоение ему характеристик

Команда управления параметрами шаблона

Перечень и описание команды управления параметрами шаблона показан на рис. 6.44. и в табл. 6.24.

Рис. 6.44. Команда управления параметрами шаблона

Команда управления параметрами шаблона

Таблица 6.24.

Обозначение	Описание
Map Profile	Профиль шаблона: назначение общих параметров, размера, образца, преобразования и проверки, данных ввода/вывода
Map flip	Переключение редактора шаблона на режимы редактирования/показа
PF-Key rules	Назначение правил значений и правил проверки ПФ - клавишам
Local data	Создание области локальных данных
Parameter Data	Создание области параметров

Установка опций редактора

Установка опций редактора шаблонов вызывается из меню Tools > Options > Map Editor (рис. 6.45.)

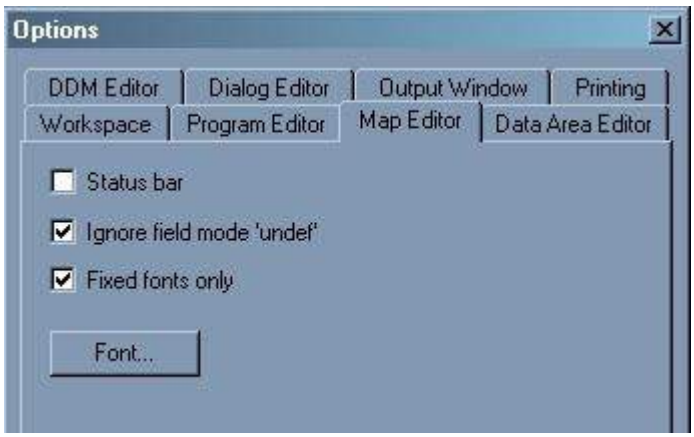


Рис. 6.45. Опции редактора шаблонов

Опции редактора шаблонов включают:

Status Bar - отображение статуса панели на верхе окна редактора.

Ignore Field Mode "Undef" - игнорирование команд CHECK или STOW.

Fixed fonts only - использование фиксированных шрифтов в диалоговом ящике "Шрифт".

Font - выбор шрифтов для шаблона (осуществляется посредством указания параметров в соответствующем окне – рис. 6.46.).

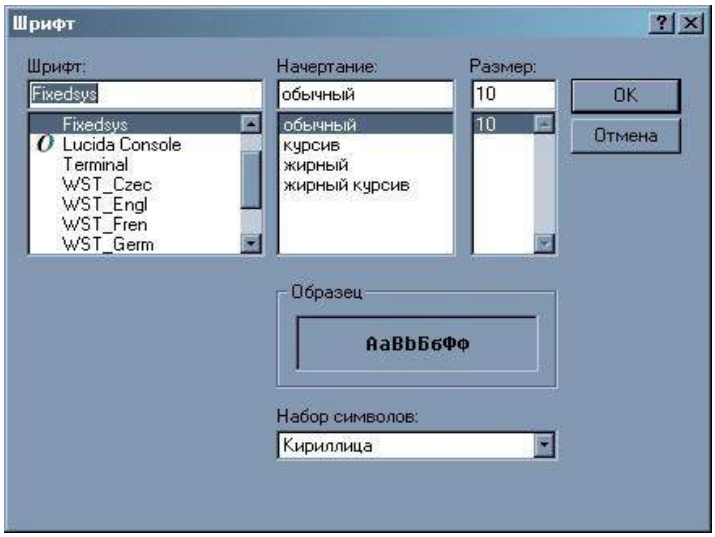


Рис. 6.46. Выбор типа шрифтов для шаблона

Модификация профиля шаблона

Команда модификации вызывается из меню Map > Map Profile (рис. 6.47.).

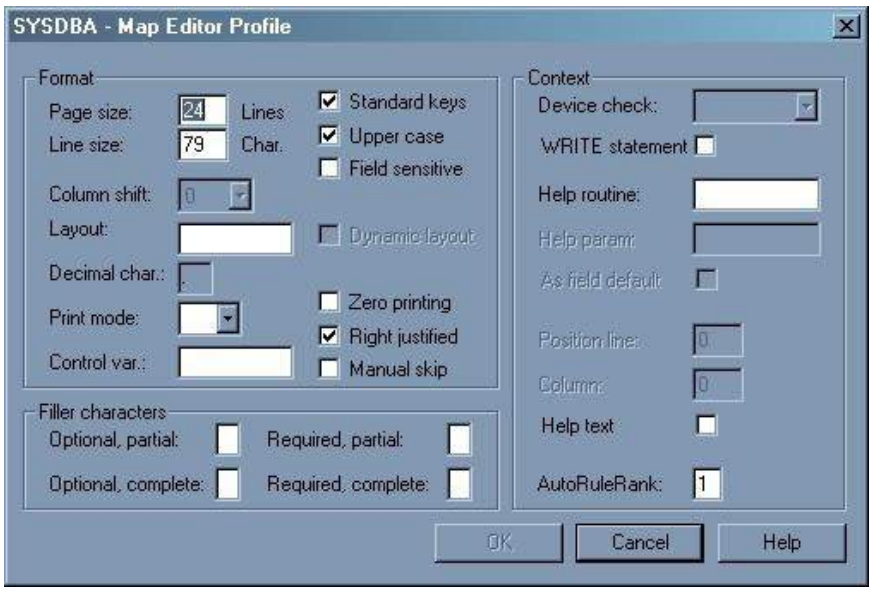


Рис. 6.47. Модификация профиля шаблона

Установка параметров профиля редактора шаблона проводится по следующим группам опций:

- Формат шаблона (Format).
- Контекстные данные (Context).
- Позиция (Position).
- Символы наполнителя (Filler character).

Опции формата шаблона показаны в таблице 6.25.

Опции формата шаблона (Format)

Таблица 6.25.

Опция	**Установка**	**Описание**
Page Size		Размер страницы, которая должна редактироваться (1 - 250). Если "Std. Keys" выбран, количество строк в диапазоне 3 - 250. Шаблон для оператора WRITE, определяет количество строк логической страницы, не размер карты. Карта может быть выходом несколько раз на одной странице.
Line Size		Номер столбцов шаблона(5 - 249).
Column Shift		Колонки переключаются (0 или 1) должно относиться к карте. Эта характеристика может использоваться, чтобы адресовать все 80 столбцов в 80-столбце экрана (Column Shift = 1, Line Size = 80).
Layout		Имя карты, которая служит в качестве стандартного формата для текущей карты
Decimal Char	"."	Символ, который должен использоваться в качестве десятичного символа. Десятичный символ может изменяться только командой Globals в библиотечном меню, или вводя

		"Globals=" команда в командной строке.
Print Mode		Способ распечатки для переменных. Эта величина копируется в определение области, когда создается новая область.
	C	Должен быть использован Альтернативный набор символов
	I	Обратное направление распечатки
	N	Стандартное направление распечатки
	Blank	Никакой способ распечатки
Standard Keys	**On**	Последние две строки карты остаются пустыми так, что функциональная ключевая спецификация может вводиться во время выполнения.
	Off	Все строки используются для карты.
Upper Case	**On**	Преобразовывается в верхний регистр
	Off	Не преобразовывается в верхний регистр
Field Sensitive	**On**	Проверка заполнения для области шаблона
	Off	Выполняется Проверка Области, когда карта заполнена полностью.
Dynamic Layout	**On**	Если Вы определили имя стандартной карты формата в "Формат" область, Вы используете эту опцию, чтобы определить, что области импортированы динамически в формат во времени компиляции. Это означает, что изменение автоматически отразится на всех аналогичных картах, использующих этот формат.
	Off	Не использовать стандартный формат динамически.
Zero Printing	**On**	Печать числовых областей, которые содержат все нули (напечатайте один нуль, выровненный по правому краю).
	Off	Подавление печати числовых областей, которые содержат все нули.
Right Justify	**On**	Числовые и текстовые области, взятые из вида пользователя или определение данных выровненные по правому краю.
	Off	Не выравниваются.
Manual Skip	**On**	Курсор не перемещается автоматически к следующей области на карте во времени выполнения, даже если бы текущая область полностью заполнена.
	Off	Курсор перемещен автоматически в следующей области на карте во времени выполнения.

Опции контекстных данных показаны в таблице 6.26.

Опции контекстных данных (Context)

Таблица 6.26.

Опция	Установка	Описание
Device Check		В этой области может рассматриваться имя устройства.
WRITE statement	**On**	Результатом является утверждением WRITE и с выводом из программы, для использования оператором WRITE USING FORM. Пустые строки в конце карты автоматически удаляются так, что карта может быть выходом несколько раз на одной странице.
	Off	Результат процесса определения карты является INPUT утверждением и результирующая карта может вводиться из программы, использующей INPUT утверждение.
Help Routine		Имя help routine, которая вызывается во времени выполнения, когда функция помощи вводится для этой карты (глобальная помощь для карты).
Help Param		Имя параметра "помощи", который вызовется во времени выполнения, когда введена функция помощи. "Помощь" параметр может только определяться, когда helproutine определен.
as field default	**On**	Файл помощи, определенный для карты прилагается как не выполняющийся в каждой индивидуальной области шаблона, что имя каждой области определено индивидуально для файла помощи. Этот выбор может только выбираться, если файл помощи определен.

Опции позиции показаны в таблице 6.27.

Опции позиции (Position)

Таблица 6.27.

Опция	Описание
Line	Позиция (строка, вертикальная) где карта помощи -, являющийся выходом.
Column	Позиция (столбец, горизонтальная) где карта помощи -, являющийся выходом.
Auto Rule Rank	Определите встроенный ранг для любого вновь созданного правила обработки на карте. (Позже, Вы можете изменить ранги индивидуально).

Опции символов наполнителя показаны в таблице 6.28.

Опции символов наполнителя (Filler Characters)

Таблица 6.28.

Опция	Описание
Optional, Partial	Указывает, что область дополнительная и может частично быть заполнена.
Optional, Complete	Указывает, что область дополнительная, но должна быть заполнена полностью, если она использована.
Required, Partial	Указывает, что требуемая область может частично быть заполнена.
Required, Complete	Указывает, что требуемая область должна быть заполнена полностью.

Назначение правил значений и проверки PF - клавишам

Описание рангов при назначении правил значений и проверки показаны на рис. 6.48. и в табл. 6.29.

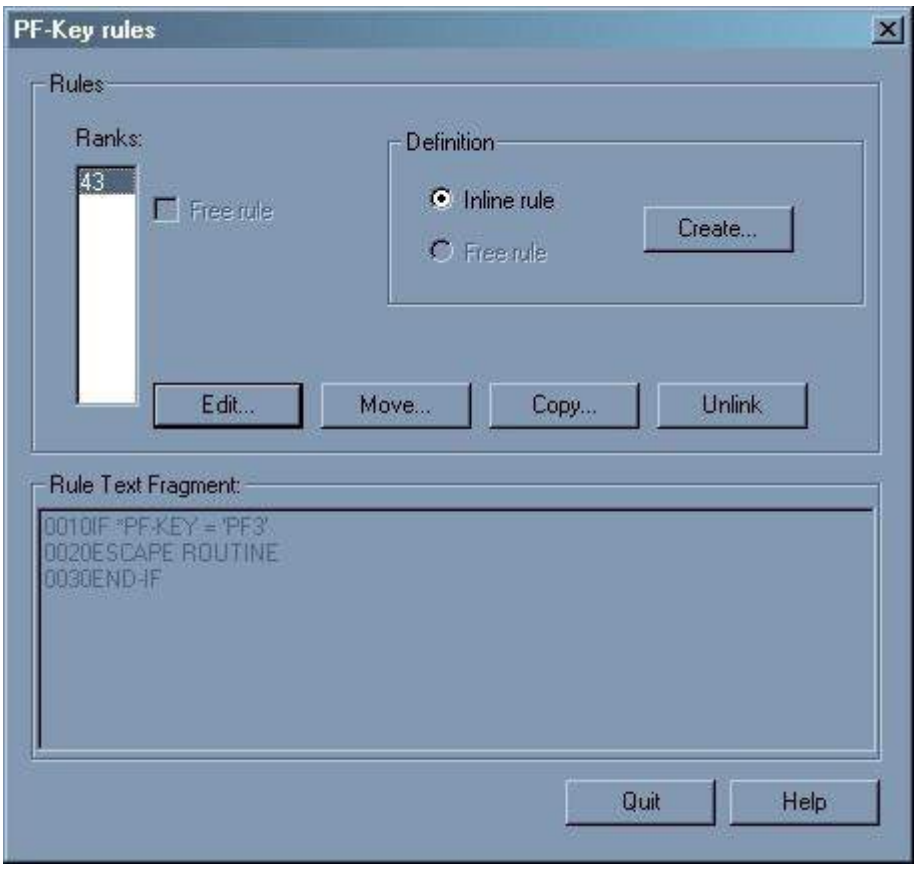


Рис. 6.48. Назначение правил значений и проверки

Назначение правил значений и обработки

Таблица 6.29.

Ранг	Правила Обработки
0	Правило Завершения
1-4	Автоматические правила
5-24	Проверка формата
25-44	Проверка значений на индивидуальные области
45-64	Перекрестный контроль между областями
65-84	Доступа к Базе Данных
85-99	Специального назначения

6.6. Свойства и возможности диалогов Natural

Диалоговый редактор

Диалоговый редактор может использоваться, чтобы создать приложение со следующими основными компонентами:

- Диалоги (Dialog(s)).
- Элементы диалога (Dialog elements).
- Атрибуты (Attributes).
- Управляемые события (Event handlers).
- Области данных (Data areas (local and parameter)).
- Встроенные подпрограммы (Inline subroutines).

Диалог представляет собой целое событийное приложение.

Создание нового диалога

Для создания нового диалогового окна, следует выбрать опцию New Dialog, панели управления ‘Object’.

Кроме того, можно отредактировать существующий диалог, выбирая его из окна отображаемой структуры файлов.

Команды редактора диалогов

Меню, кнопки панели, и команды пригодные для диалогового редактора могут использоваться, чтобы создать компоненты событийного приложения и отредактировать их в различных окнах редактора. Можно создать или отредактировать другой диалог, или вводить другой редактор и создавать или

редактировать другой тип объекта (например, программа, DDM или область данных).

Команды работы с объектом

Оисание команд работы с объектом показаны на рис. 6.51. и в табл. 6.30.

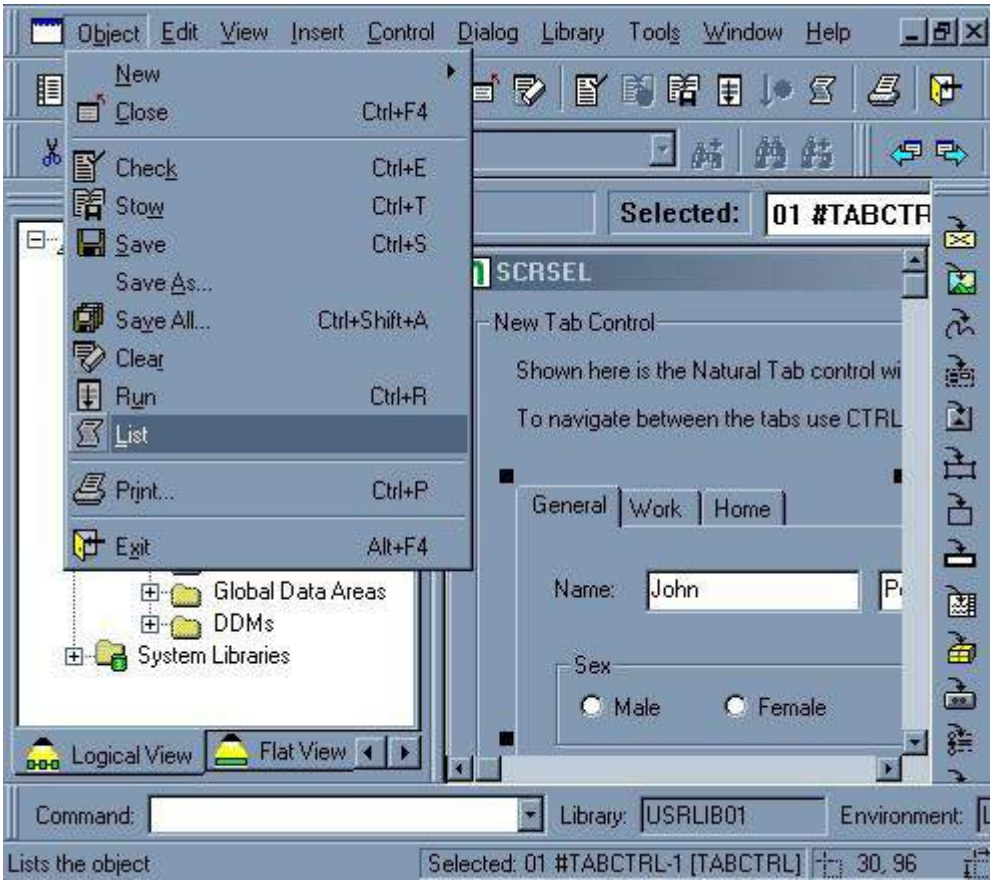


Рис. 6.51. Команды работы с объектом

Команды работы с объектом

Таблица 6.30.

Обозначение	**Описание**
New	Создать новый объект: программы, области, и т.д.
Close	Закрыть рабочую область
Check	Проверить содержимого редактора на синтаксические ошибки
Save	Сохранить содержимое редактора в исходом коде
Save As …	Сохранить содержимое редактора в другой библиотеке или с другим именем и типом
Save All …	Сохранить все объекты в исходном коде
Clear	Очистить рабочую область
Run	Выполнить программу из содержимого редактор
List	Просмотреть расширенный листинг объекта
Print	Распечатать содержимое на принтере
Exit	Выйти из редактора и сеанса Natural

Команды редактирования объекта

Перечень и описание команд редактирования объекта показаны на рис. 6.52. и в табл. 6.31.

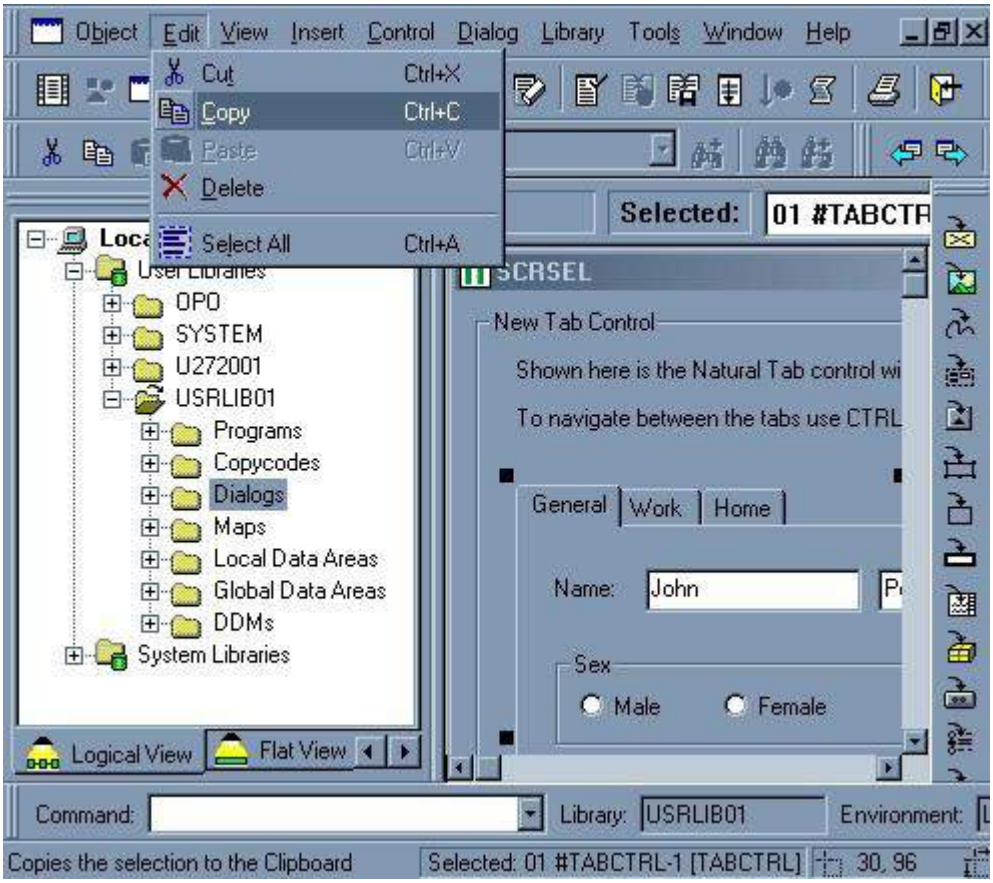


Рис. 6.52. Команды редактирования объекта

Команды редактирования объекта

Таблица 6.31.

Обозначение	**Описание**
Cut	Вырезать
Copy	Скопировать
Paste	Вставить
Delete	Удалить
Select All	Выбрать все элементы

Функции команды вставки элементов

Перечень функций команды вставки элементов показан на рис. 6.53.

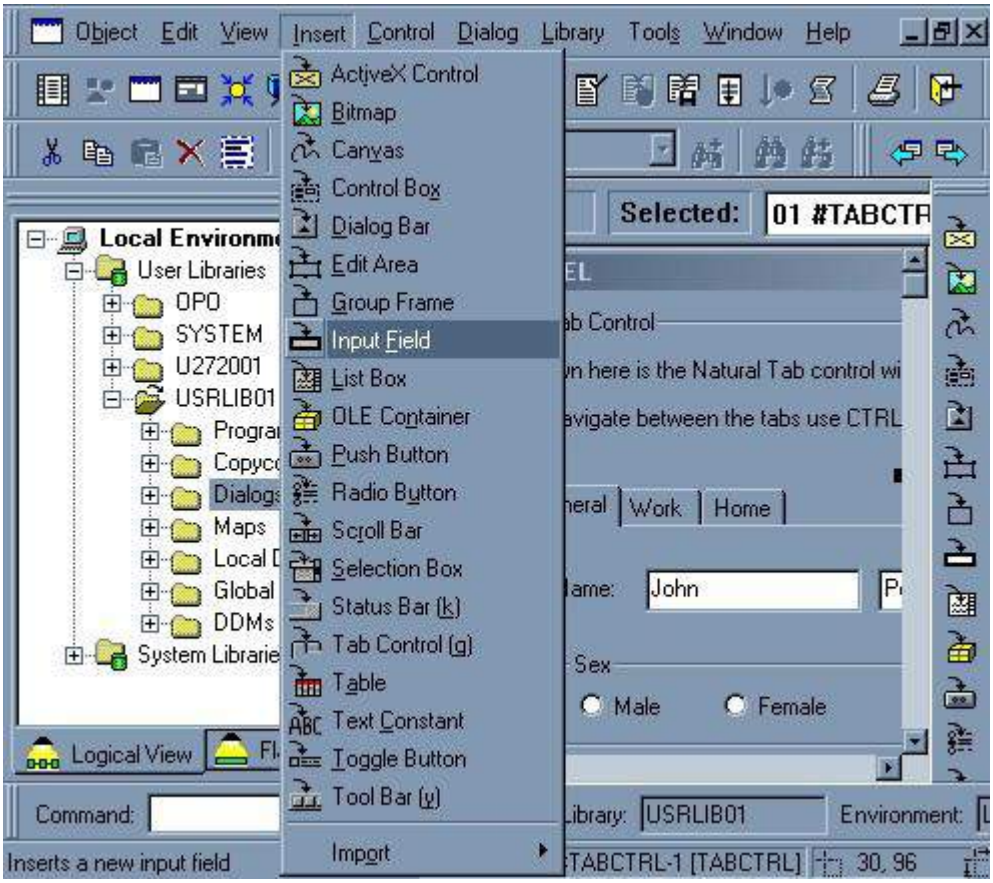


Рис. 6.53. Функции команды вставки элементов

Функция вставки элементов меню "Insert", позволяет выбирать и поместить в тело диалога следующие элементы:

- ActiveX управление (ActiveX Control);
- изображение (Bitmap);
- канва (Canvas);
- управляющий ящик (Dialog Bar);
- область редактирования (Edit Area);
- групповой фрейм (Group Frame);
- входная область (Input Field);
- список (List Box);
- контейнер OLE (OLE Container);
- кнопка (Push Button);
- переключатель Radio (Radio Button);

- зона прокрутки (Scroll Bar);
- ящик выбора (Selection Box);
- таблица (Table);
- текстовая константа (Text Constant);
- кнопка переключателя (Toggle Button).

Для импортирования определение полей из других объектов следует воспользоваться опцией Import …

Меню атрибутов управления

Перечень и описание меню атрибутов управления показаны на рис. 6.54. и в табл. 6.32.

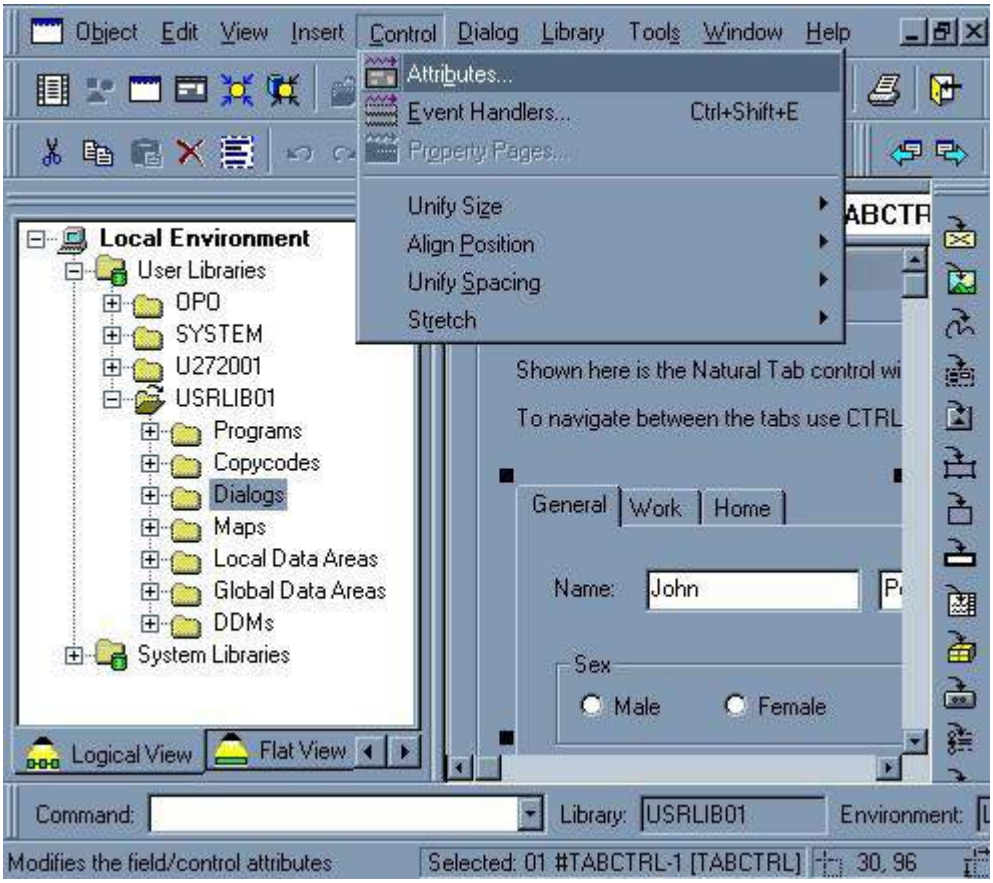


Рис. 6.54. Меню атрибутов управления

Меню атрибутов управления

Таблица 6.32.

Обозначение	Описание
Attributes	Редактирование атрибутов диалоговых элементов управление событиями
Event Handlers	Редактирование диалоговых элементов управления событиями
Property Pages	Свойства страницы
Unify size	Объединение размера различных диалоговых элементов
Align position	Выравнивание позиции различных диалоговых элементов
Unify spacing	Объединение расстояния между различными диалоговыми элементами
Stretch	Растягивать размер элемента по разным направлениям

Редактирование диалогов

Перечень и описание опций редактирования диалогов показаны на рис. 6.55. и в табл. 6.33

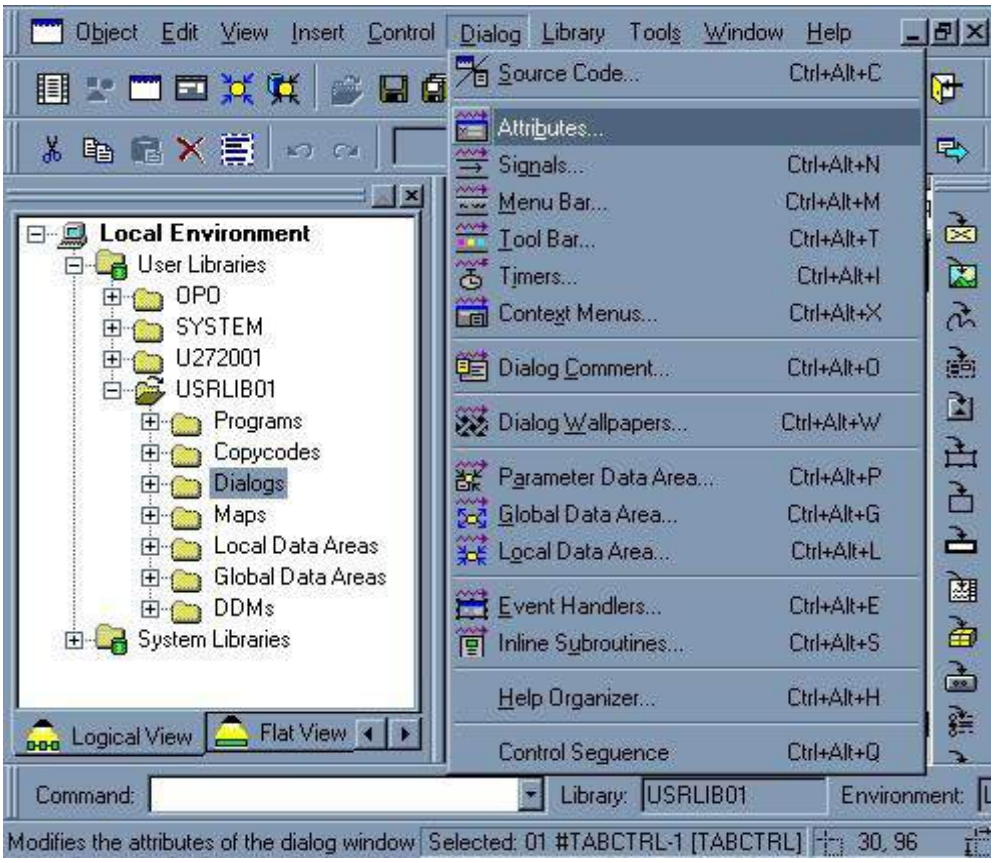


Рис. 6.55. Опции редактирования диалогов

Опции редактирования диалогов

Таблица 6.33.

Обозначение	Описание
Source code	Редактирование текста программы диалога редактором программ
Attributes	Редактирование атрибутов диалога
Signals	Создание и поддержка сигналов для диалога
Menu Bar	Редактирование зоны меню на верху диалогового окна. Это - верхний уровень структуры меню и содержит пункты меню
Tool bar	Редактирование зоны инструментария диалогового окна
Times	Создание и поддержка таймера для диалога
Context Menus	Определение контекстное - специфическое меню
Dialog Comment	Добавление раздела комментария к диалогу
Dialog Wallpapers	Создание и поддержание обоев
Parameter Data Area	Определение параметра или локальная область данных для диалога
Local Data Area	Выбор локальной области данных для диалога
Global Data Area	Выбор глобальной области данных для диалога
Event Handlers	Редактирование обработчика случаев
Inline Subroutines	Определение встроенной подпрограммы для диалога
Help Organizer	Определение и поддержание инструкции для диалога
Control Sequence	Определение управляющей последовательности в диалоге

Окно редактора диалога

Общий вид окна редактора диалога показан на рис. 6.56.

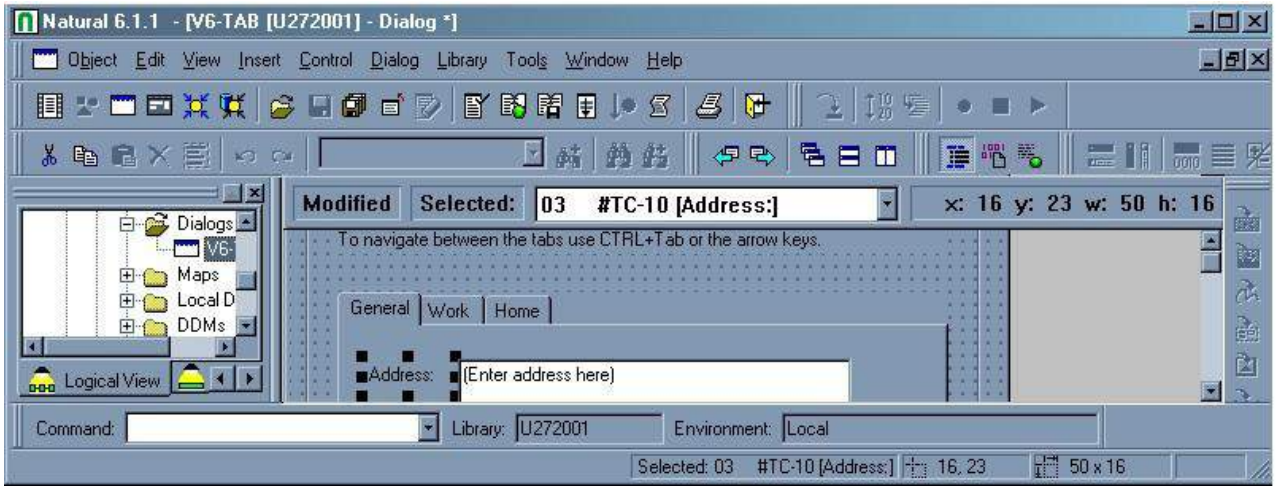


Рис. 6.56. Окно редактора диалога

Заголовок окна редактора диалога содержит следующую информацию (табл. 6.34.)

Окно редактора диалога

Таблица 6.34.

Пункт	Объяснение
Status	Указывает состояние модификации диалога (статус)
Selected (handle)	Указывает имя выбранного диалогового элемента; отображает ящик выбора, отображает элементы диалога вместе с их номером уровня в диалоговой элементной иерархии. Можно выбрать другой диалоговый элемент в ящике выбора
x	Позиция ось X выбранного диалогового элемента относительно верхнего левого угла области клиента родительского диалогового элемента (или диалог, для диалоговых элементов верхнего уровня). Эквивалентный текущей величине ПРЯМОУГОЛЬНИКА-X атрибут
y	Позиция ось Y выбранного диалогового элемента относительно верхнего левого угла области клиента родительского диалогового элемента (или диалог, для диалоговых элементов верхнего уровня). Эквивалентный текущей величине ПРЯМОУГОЛЬНИКА-Y атрибут
w	Ширина выбранного диалогового элемента. Эквивалентный текущей величине ПРЯМОУГОЛЬНИКА-W атрибут
h	Высота выбранного диалогового элемента. Эквивалентный текущей величине ПРЯМОУГОЛЬНИКА-H атрибут

Начальные установки размеров экрана диалога

Установки размеров экрана диалога, предлагаемые по умолчанию, показаны на рис. 6.57.

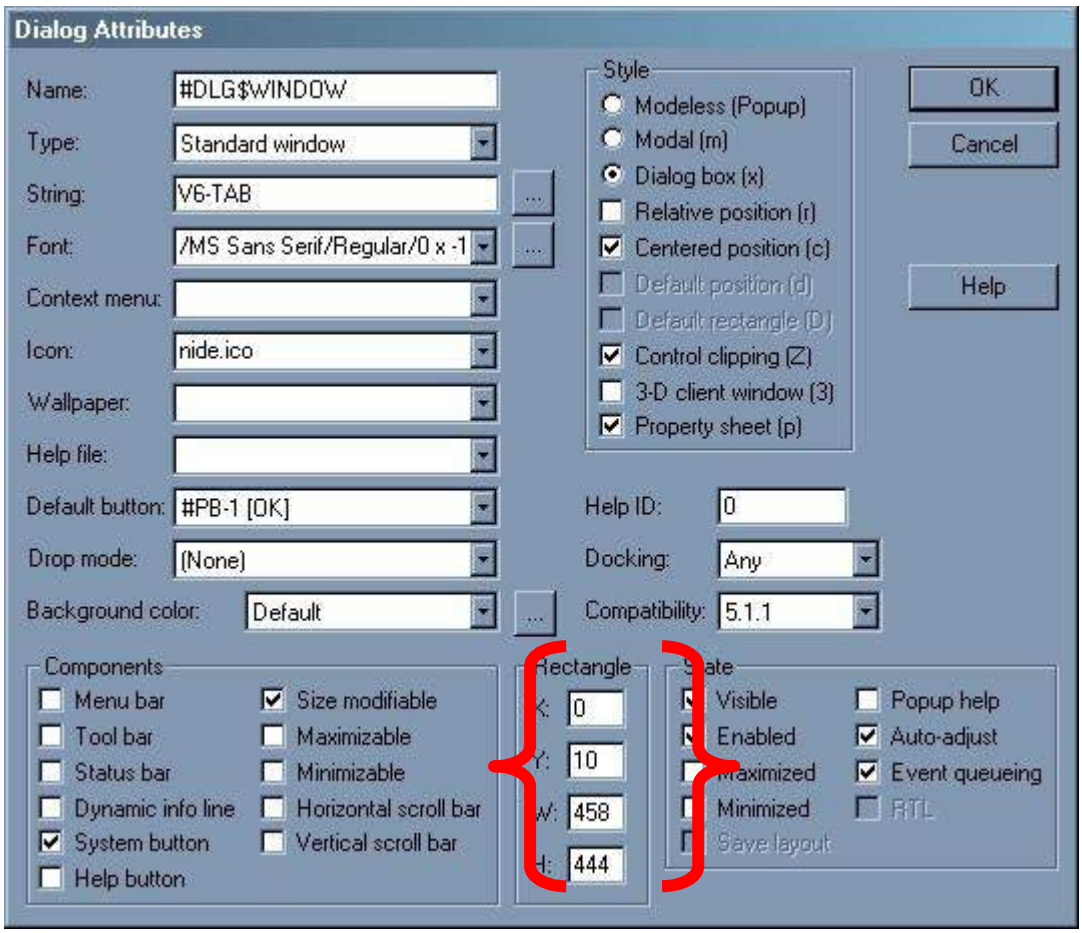


Рис. 6.57. Начальные установки размера экрана диалога

Для изменения *начальной позиции* диалога следует выполнить одну из следующих операций:

- Открытие окна атрибутов и занесение новых координат (в пикселях) в "X" и "Y" области.

- Выбор опции ‘Attributes’ через меню ‘Dialog’ или из диалогового контекстного меню и изменение параметров “X” и “Y”.

Для изменения *начального размера* диалога следует:

- Также использовать калибровку границы диалога.

- Открыть свое окно атрибутов и занести новый размер (в пикселях) в "W" и "H" области.

Использование команд перемещения элементов

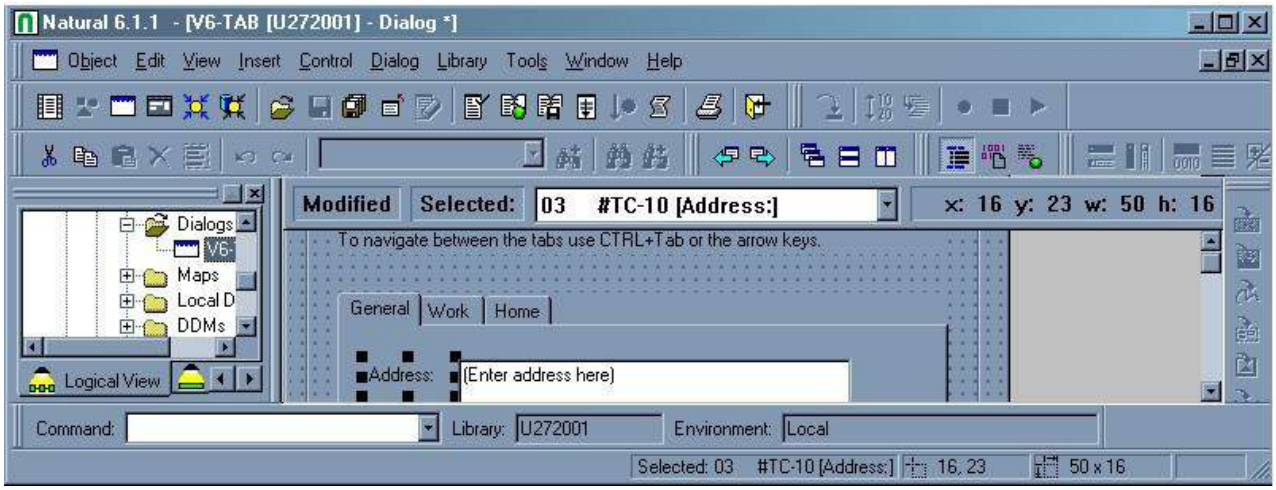


Рис. 6.58. Использование команд перемещения элементов

В рамках использования команд перемещения элементов (рис. 6.58.) предлагается использовать следующие операции:

- Выбор и отмена выбора.
- Перемещение элементов.
- Изменение размера диалогового элемента.
- Перемещение указателя.

Выбор и отмена выбора

Возможно, выбирать многочисленные диалоговые элементы, но только один может быть активным в конкретное время. Активный выбор описывается черными маркерами выбора, использующими какое управление, может менять размеры. Неактивный выбор описывается серыми маркерами выбора.

Любое функционирование, которое завершается версией левой кнопки мыши, может прерываться нажатием клавиши ESC прежде, чем выпустить левую кнопку мыши.

Перемещение элементов

Выбранные элементы можно перемещать стрелками курсора клавиатуры:

- на 8 пикселей, удерживая клавишу SHIFT;
- на 1 пиксель, удерживая клавиши SHIFT+CTRL.

Изменение размера диалогового элемента

Изменение размера можно осуществлять несколькими способами:

- Растягивая мышью границы элемента.
- Открывая атрибуты и задавая высоту (H) и ширину (W).
- Через команду из меню Control > Stretch (протянуть).

Перемещение указателя

Перемещение указателя осуществляется тремя способами:

- Двигая мышью.
- Нажимая любая клавишу перемещения курсора, чтобы переместить указатель на количеством пикселей определенных в ‘Options Dialog Editor Grid Settings'.
- Удерживая CTRL и нажимая любую клавишу перемещения курсора, чтобы переместить указатель на 1 пиксель.

Перечень *стандартных клавиш*, используемых при перемещении элементов, показаны в таблице 6.35.

Стандартные клавиши

Таблица 6.35.

Клавиша (комбинация)	**Функция**
DEL	Удаление выбранного диалогового элемента
CTRL+C	Копирование элемента
CTRL+V	Вставка элемента

Организация помощи

Организация помощи вызывается из меню Dialog > Help Organizer (рис. 6.59.)

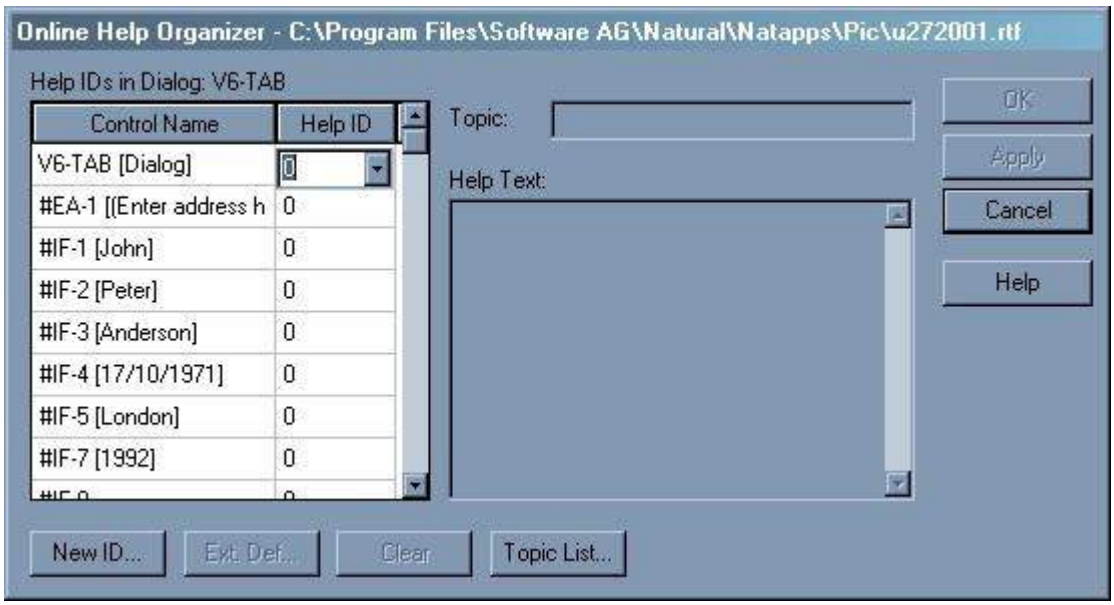


Рис. 6.59. Организация помощи

Организатор помощи позволяет:

- назначать помощь ID (Help-ID) в специфическом диалоговом элементе;
- написать инструкцию для этой темы помощи; текст преобразовываете в файл RTF, который должен обрабатываться компилятором подсказки;
- определять тему подсказки ключевых слов;
- назначать макро компилятор подсказки в тему подсказки;
- добавлять комментарий для ваших внутренних целей документации.

Можно также использовать один и тот же файл подсказки для нескольких диалогов.

Файл .RTF и соответствующая информация темы подсказки, создаваемые организатором подсказки должны преобразовываться в файл .HLP. Таким образом, конечный пользователь может извлечь справочную информацию из приложения Natural. Это преобразование в файл .HLP делается компилятором подсказки.

Опции редактора диалога

Опции редактора диалога вызывается из меню Tools > Options > Dialog Editor (рис. 6.60.)

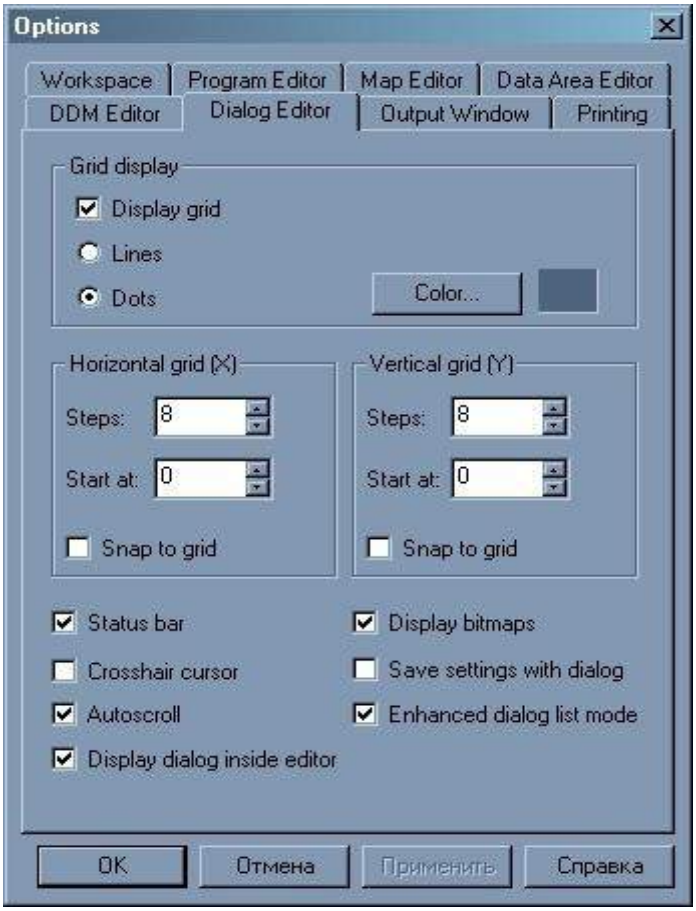


Рис. 6.60. Опции редактора диалога

Опции редактора диалога включают:

Status Bar - отображение или скрытие панели статуса выполнения команд.

Crosshair Cursor - по умолчанию, диалоговый редактор отображает системный курсор. Если включается, тогда можно создавать или перемещать диалоговые элементы на расширенных границах окна.

Auto scroll - проверяется кнопка переключателя и диалоговое окно редактора автоматически перемещается, чтобы сделать диалог видимым, если диалог перемещен частично за пределами текущей видимой области.

Display dialog inside editor - диалоговый редактор позволяет определять, отображается ли текущий диалог в диалоговом окне редактора (умолчание). Отображение диалога за пределами пространства для редактирования. Нужно

быть внимательным, устанавливая значение как «OFF», т.к. диалог может исчезнуть из вида.

Display bitmaps - по умолчанию, диалоговый редактор загружает и отображает графические объекты/изображения в диалоговых элементах. Если Вы выключаете этот выбор, диалоговый элемент отображает имя объекта.

Display Grid - диалоговый редактор позволяет отображать сетку, чтобы помочь выравнивать диалоговые элементы. По умолчанию сетка не отображается.

Save settings with dialog - выборы будут сохранены вместе с диалогом всякий раз, когда любой диалог сохранен. Диалоги, сохраненные при неактивном выборе, используют параметры по умолчанию.

Enhanced dialog list mode - установка просмотра расширенного диалогового списка.

7. СОСТАВ И ИНСТРУМЕНТАРИЙ ЕДИНОЙ СРЕДЫ РАЗРАБОТКИ ПРИЛОЖЕНИЙ SPOD

SPoD (Single Point of Development) – единая среда разработки прикладных приложений, позволяющая конструировать приложения для различных платформ (Windows, Unix, Mainframe), для различных СУБД (Adabas, Tamino, Oracle, DB2 и пр.), а также с использованием различных технологий (DCOM, SOAP, Web Services, и пр.)

SPoD удовлетворяет следующим требованиям и предлагает следующие преимущества:

−Клиент/сервер архитектуры, поддерживающий одну единственную удаленную среду разработки для всех платформ.

−Единый, знакомый образ всех объектов включаемых в прикладную разработку.

−Расширенная управляемая помощь/документация для дистанционных сред разработки.

−Увеличение производительности разработки благодаря мощной графической рабочей среде.

−Управление над заданиями разработки.

−Низкие издержки для разработки программного обеспечения, эксплуатации и администрирования.

7.1. Архитектура среды SPoD

Технология разработки приложений SPoD использует редактор Natural Studio в среде Microsoft Windows в качестве клиента разработки, Natural Development Server и Natural на серверной платформе, а также TCP/IP в качестве транспортного протокола. Архитектура системы SPoD показана на рис. 5.2.

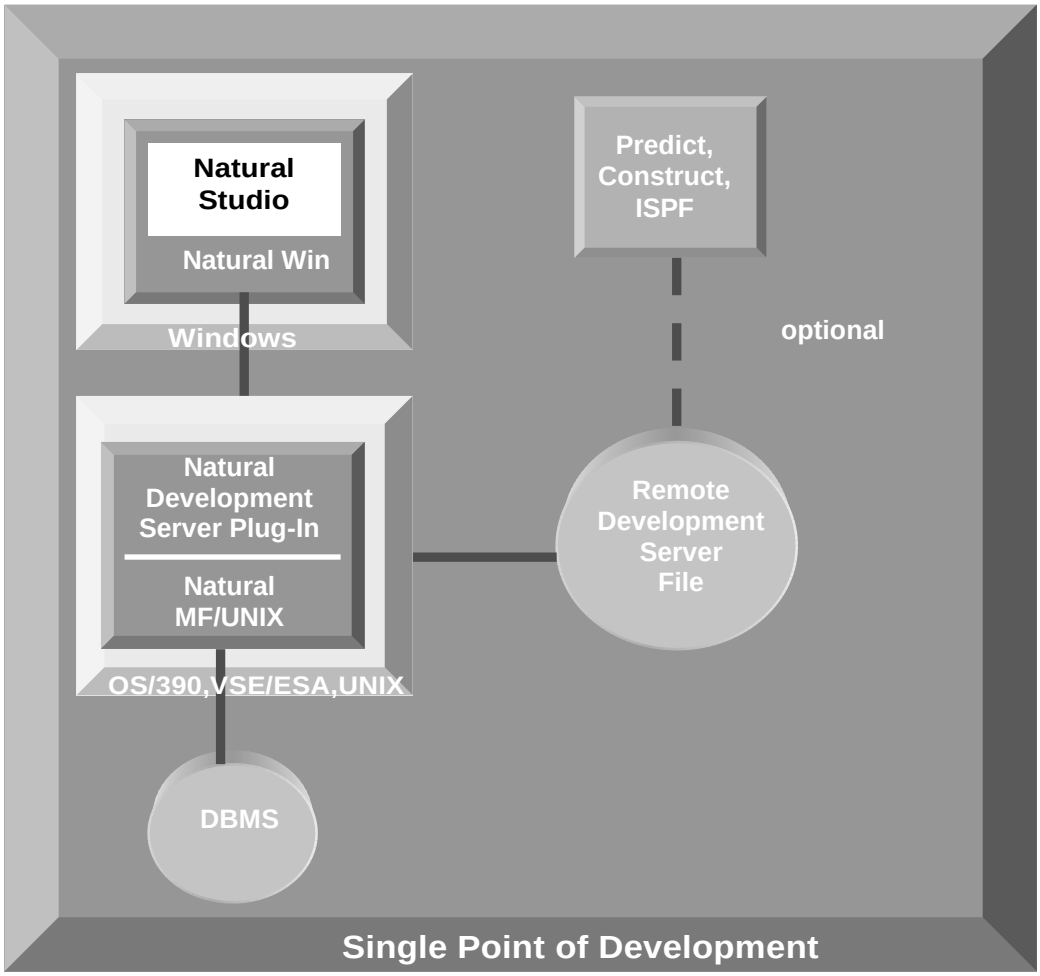


Рис. 7.1. Архитектура системы SPoD

Система SPoD включает следующие средства:

– Natural для Windows – клиент удаленной разработки;

– Natural Development Server – сервер разработки;

– Development Server File – файл хранения данных;

– Database Management System – система управления базами данных;

– Predict and Natural Construct – идентичность файла хранения данных.

Natural для Windows – клиент удаленной разработки. При использовании в качестве клиента удаленной разработки, Natural Studio является рабочей станцией центральной разработки для всех остальных платформ. Все Natural-связанные прикладные разработки, включая шаги конфигурации, можно осуществлять из среды Natural Studio, независимо от платформы, на которой находится серверная среда разработки. Большинство функций, необходимых для удалённой разработки приложений, таких как администрирование, удаленное редактирование, компиляция, отладка, и т.п., интегрировано в Natural Studio.

Natural Development Server – сервер разработки. Используется для удаленной разработки на серверной платформе. Сервером разработки является Natural вместе с добавленным и установленным продуктом Natural Development Server (NDV).

Использование удаленного сервера разработки обеспечивает следующие функции:

−предоставляет доступ к системным файлам;

−обеспечивает доступ к прикладным данным;

−гарантирует согласованность модификаций прикладных объектов;

−выполняет дистанционные команды, задаваемые клиентом разработки.

Development Server File – файл хранения данных. Файл сервера разработки (Development Server File) хранит прикладные данные.

Database Management System – система управления базами данных. Дистанционная среда разработки Natural может использоваться с любым компонентом, поддерживающим систему управления базы данных, которая доступна для серверной платформы и выполняется под существующей операционной системой. В целях обеспечения максимальной производительности рекомендуется использовать СУБД ADABAS и Tamino, которые, как и система SPoD, являются продуктами фирмы Software AG.

Predict and Natural Construct – идентичность файла хранения данных. Predict является встроенным словарем данных, основная функция которого – автоматическая генерация программного кода на основе свойств задаваемых объектов. Natural Construct – это генератор прикладных систем, предназначенный для обеспечения гибкости структуры пользовательских приложений.

7.2. Функции и свойства среды SPoD

Основными функциями среды SPoD являются:

−Определение и генерация базы данных (ADABAS, или иная профессиональная СУБД).

−Разработка программ с использованием сервисов систем обработки транзакций (CICS, Complete).

−Генерация программ.

−Документирование приложений.

−Реинжениринг и поддержка приложений.

−Работа в среде Natural Studio с объектами операционной системы (очередями заданий, наборами данных и т.д.).

Основными свойствами среды SPoD являются:

−Дистанционная обработка объекта. Т.е. возможность манипулирования программными объектами, независимо от физического размещения.

−Дистанционное редактирование. Возможность извлечения исходных файлов, размещенных на сервере и редактирования их на рабочей станции пользователя с последующим сохранением их в месте физического размещения.

−Дистанционная компиляция. В случае компиляции с рабочей станции пользователя, исполнение команды на сервере.

−Удаленная отладка. Возможность отладки приложений, выполняемых на сервере, которая может располагаться в той же системе или в универсальных Natural средах сервера.

−Информация перекрестных ссылок. Тиражирование информации об объектах и их ссылках сохраняется в файл сервера.

−Блокировка объектов. В случае доступа к дистанционному серверу разработки, предотвращение параллельной коррекции посредством блокировки. Блокировка информации держится в файле сервера разработки.

−Поддержка окна эмуляции терминалов. Эксплуатация универсальных приложений часто включает испытание выхода на терминалы. Для этой цели окно эмуляции терминалов выдается автоматически.

7.3. Компоненты среды SPoD

Рабочее место разработчика в среде Microsoft Windows использует встроенную систему Natural Studio со следующими компонентами:

−Program Editor – редактор программ.

−DDM Editor – редактор DDM.

−Data Editor – редактор областей данных.

−Map Editor – редактор шаблонов.

−Dialog Editor – редактор диалога.

– Navigator – средство поиска и управления объектами.

– Debugger – отладчик программ.

– Natural Data Browser – средство просмотра данных в БД.

– Component Browser – просмотр объектов библиотек DCOM.

– Natural Reporter – средство составления отчетов.

В качестве дополнительных компонент могут быть активизированы другие средства разработки:

– Predict - словарь данных.

– Natural Construct - генератор прикладных систем.

– Natural Engineer - средство реинжиниринга приложений.

– Natural ISPF - встроенное структурное средство программирования.

Natural Studio включает ряд специализированных редакторов, каждый из которых отвечает за определенную функцию, что способствует эффективной и удобной разработке и модификации программных объектов Natural. Эти редакторы, а также средства просмотра библиотек и прочие инструменты объединены в интегрированную систему, называемую *Natural Studio*.

Program Editor используется для редактирования программ, подпрограмм, вложенных процедур, копикодов и текстов, разрабатываемых средствами Natural.

DDM Editor - редактор описаний таблиц баз данных (Data Definition Modules).

Data Editor - редактор описаний областей данных, таких как Local Data Area, Global Data Area, Parameter Data Area и Object Data Area. Области данных допускают использование в различных программных объектах.

Map Editor - редактор разработки экранов. Экраны Natural могут содержать фиксированный текст, области ввода и области вывода, а также ограниченный набор элементов графики, таких как меню, кнопки переключения, поля прокрутки и изображения. При проектировании панели допускается импортировать поля из других программных объектов Natural, что уменьшает разночтения в написании имен переменных и определений типов.

Dialog Editor - редактор разработки стандартизированных GUI-диалогов, которые управляются событийно-ориентированными средствами Natural.

Также редактор может быть использован как редактор пользовательских форм и панелей.

Navigator – обеспечивает поиск и управление программными объектами Natural в пределах библиотек. Кроме того, Navigator предлагает функции массовой обработки, рассчитанные на целые наборы программных объектов.

Debugger - развитая система символьной отладки, которая дает пользователям возможность анализировать программу по шагам, расставлять точки прерывания и наблюдать за изменениями значений переменных. Отладчик может работать также в удаленном режиме.

Natural Data Browser обеспечивает быстрый доступ к данным ADABAS. Позволяет просматривать данные ADABAS для проверки, выбирать файлы и поля, генерировать отчеты.

Component Browser дает пользователям возможность просматривать библиотеку объектов для получения описания интерфейсов объектов, элементов управления или объектов автоматизации, для их использования в программе Natural. Component Browser автоматически создает код образца, который готов к копированию в программу и к немедленному использованию без выполнения явной процедуры импорта.

Natural Reporter - полнофункциональное средство составления отчетов, совместимое с любой СУБД, поддерживаемой ODBC. Natural Reporter позволяет создавать отчеты, в которых объединены данные, тексты, графики и рисунки. Редактором поддерживаются различные варианты шрифтов, обрамления и затенения. Предлагаются и специальные функции, в числе которых сортировка, выбор по критерию, поля формул, верхние и нижние колонтитулы для одной или нескольких страниц, разрывы группировок страниц, подсчет итогов в группе или во всем документе.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. An analysis of questions: preliminary report, System Development Corporation, Santa Monica, California, 1963.
2. CODASYL DBTG Report, "Feature Analysis of Generalized Data Base Management Systems", 1971, (Русск. пер.: Информационные системы общего назначения. Пер. с англ. - М.: Статистика, 1975).
3. Codd E.F. A Relational Model of Data for Large Shared Data Banks. Comm. of the ACM, 1970, v.13, no. 6, pp.377-387. (Русск. пер.: Кодд Е.Ф. Реляционная модель для больших совместно используемых банков данных // СУБД. – 1995. - №1. – С. 145-160)
4. CODASYL DDLC Report, "CODASYL Database Languages and the ANSI/SPARC Framework", 1978.
5. CODASYL Systems Committee. "A Survey of Generalized Data Base Management Systems". – Technical Report, 1969.
6. Зайцев С.Л. Автоматизированные системы управления предприятиями стандарта ERP/MRP II. М.:2001
7. Автоматизация систем управления предприятиями стандарта ERP-MRPII. / Обухов И.А., Гайфуллин Б.Н. - Интерфейс-Пресс, 2002.
8. Jardine D. Data Base Management Systems. Montreal, Canada, 1974.
9. Palmer J. Data Base Systems: A Practical Reference. – Q.E.D. Information Sciences Inc. Massachusetts, 1975.
10. Codd E. Normalized Data Base Structure. A Brief Tutorial. – Proc. 1971 ACM SIGFIDET Workshop on Data Description, Access and Control.
11. Engles R. A Tutorial on Data Base Organization. - Annual Review in Automatic Programming, New York, 1972.
12. Bernstein P., Swenson J., Tsichritzis D. A Unified Approach to Functional Dependencies and Relations. – Proc. 1975 ACM SIGMOD International Conference on Management of Data.

13. Дейт К. Введение в системы баз данных: Пер. с англ. – СПб.: Вильямс, 1999. – 464 с.
14. Lefkovitz D. File Structures for On-Line Systems. – Spartan Books, 1971.
15. Lum V. Multi-attribute Retrieval with Combined Indexes. CACM, 1970.
16. Мартин Дж. Организация баз данных в вычислительных системах: Пер. с англ. / Под ред. А.А. Стогния и А.Л. Щерса. – М.: Мир, 1980. – 672 с.
17. Мейер Д. Теория реляционных баз данных: Пер. с англ. / Под ред. М.Ш. Цаленко. – М.: Мир, 1987.
18. Senko M., Altman E., Astrahan M., Fehder P. Data Structures and Accessing in Data-Base Systems. – IBM System J., v. 12, №1, 1973
19. Bayer R., McCreigth E.M. Organization and maintenance of large ordered indicates. Acta Informatica, 1:3; 1972, pp. 173-189.
20. T.B. Pedersen, C.S. Jensen, C.E. Dyreson, «A Foundation for Capturing and Querying Complex Multidimensional Data», Information Systems, vol. 26, no. 5, 2001
21. T.B. Pedersen, C.S. Jensen, C.E. Dyreson, «Extending Practical Pre-Aggregation in On-Line Analytical Processing», Proc. 25th Int'l Conf. Very Large Databases, Morgan Kaufmann, San Mateo, Calif., 1999
22. DeWitt, D.J., Katz, R., Olken, F., Shapiro, D., Stonebraker, M. and Wood, D. Implementation techniques for main memory database systems. In Proceedings of the 1984 SIGMOD Conference, Boston, 1984.
23. Верников Г. Г. Стандарт MRPII. Структура и основные принципы работы MRPII-систем. – М: 2002.
24. Верников Г. Г. Философия и основные понятия MRP, - М: 2002.
25. Коннолли Т., Бегг К., Страчан А. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 2-е изд.: Пер. с англ. – М.: Вильямс, 2000.
26. Хансен Г., Хансен Дж. Базы данных: разработка и управление: Пер. с англ. – М.: БИНОМ, 1999.

27. J. Gray et al., «Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals», Data Mining and Knowledge Discovery, vol. 1, no. 1, 1997
28. R. Kimball, The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses, John Wiley & Sons, New York, 1996
29. Nievergelt J., Hinterberger H., Sevchik K.S., The grid file: an adaptable symmetric multikey file sructure. ACM Trans. on Database Systems, v. 9, no. 1, 1984.
30. Gutman A. R-trees: A Dynamic Index Structure for Spatial Searching. Proc. ACM SIGMOD Conf. on Management of Data. Boston, 1984.
31. Sellis T., Roussopoulos N., Faloutsos C. The R+-Tree: A Dynamic Index for Multi-Dimensional Objects. Proc. of Int. Conf. on Very Large Data Bases, Brighton, 1987.
32. Beckman N., Krigel H.P., Schneider R., Seeger B. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. Proc. ACM SIGMOD Conf. on Management of Data. New Jersey, 1990.
33. Samet H. The Quadtree and Related Hierarchical Data Structures. ACM Computing Surveys, v. 16, no. 2, 1984.
34. Robinson J. The K-D-B Tree: A Search Structure for Large Multidimensional Dynamic Indexes. Proc. of ACM SIGMOD Conf. of Management of Data, Ann Arbor, 1981.
35. P. Vassiliadis, T.K. Sellis, «A Survey of Logical Models for OLAP Databases», ACM SIGMOD Record, vol. 28, no. 4, 1999
36. E. Thomsen, OLAP Solutions: Building Multidimensional Information Systems, John Wiley & Sons, New York, 1997
37. H.-J. Lenz, A. Shoshani, «Summarizability in OLAP and Statistical Data Bases», Proc. 9th Int'l Conf. Scientific and Statistical Database Management, IEEE CS Press, Los Alamitos, Calif., 1997

38. T. Zurek, M. Sinnwell, «Data Warehousing Has More Colours Than Just Black and White», Proc. 25th Int'l Conf. Very Large Databases, Morgan Kaufmann, San Mateo, Calif., 1999
39. Глинских А.А. Мировой рынок ERP-систем. Jet Info - информационный бюллетень. 2002. №2 (105).
40. Крол Эд. Все об Internet. Руководство и каталог:Пер. с англ. Киев: BHV, 1995.
41. Date C.J. What is distributed database? InfoDB, 2:7. 1987.
42. Флореску Д., Леви А., Мендельсон А. Технологии баз данных для World-Wide Web: обзор: Пер. с англ. // СУБД. 1998. - №4-5.
43. Базы данных: достижения и перспективы на пороге 21-го столетия: Пер. с англ. / Под ред. А. Зильбершатца, М. Стоунбрекера и Дж. Ульмана // СУБД, №3 – 1996
44. Введение в XML: Пер. с англ. / Кэгл К., Гиббонс Д., Хантер Д., Озу Н., Пиннок Д., Спенсер П. - М.: Лори, 2001. - 656 с.
45. Воронов М.П. Информационные технологии в управлении: СУБД ADABAS и проектирование приложений средствами NATURAL / М.П. Воронов, А.С. Фатеркин, В.П. Часовских // Екатеринбург: Уральский государственный лесотехнический университет, 2006. - 477 с.
46. Старыгин А. XML. Разработка Web-приложений. – СПб.: BHV-СПб, 2003. – 592 с.
47. Автоматизированные информационные технологии в экономике: Учебник/М. И. Семенов, И. Т. Трубилин, В. И. Лойко, Т. П. Барановская; Под общ. ред. Трубилина. – М.: Финансы и статистика, 2000. – 416 с.
48. Алиев Р.А., Абдикеев Н.М., Шахназаров М.Н. Производственные системы с искусственным интеллектом. – М.: Радио и связь, 1990. – 264 с.
49. Аппак М.А. Базы данных в АСУ – связь. – М.: Радио и связь, 1987. – 80 с.
50. Воронов М.П. Построение жестко фиксированной модели структуры организации средствами ADABAS и Natural при внедрении интегрированной системы управления / М.П. Воронов, М.П. Железняк,

В.В. Томилин // Материалы всероссийской научно-технической конференции студентов и аспирантов: Материалы всероссийской научн.-техн. конф. – Екатеринбург: Уральский государственный лесотехнический университет, 2005. - С. 235.

51. Воронов М.П. Построение гибкой модели структуры организации средствами ADABAS и Natural при внедрении интегрированной системы управления / М.П. Воронов, М.П. Железняк, В.В. Томилин // Материалы всероссийской научно-технической конференции студентов и аспирантов: Материалы всероссийской научн.-техн. конф. – Екатеринбург: Уральский государственный лесотехнический университет, 2005. - С. 236.
52. Воронов М.П. Построение смешанной модели структуры организации средствами ADABAS и Natural при внедрении интегрированной системы управления / М.П. Воронов, М.П. Железняк, В.В. Томилин // Материалы всероссийской научно-технической конференции студентов и аспирантов: Материалы всероссийской научн.-техн. конф. – Екатеринбург: Уральский государственный лесотехнический университет, 2005. - С. 237.
53. Воронов М.П. Особенности автоматизации документооборота в ВУЗе с использованием средств Software AG (ADABAS, Natural, Entire NetWork) / М.П. Воронов, К.К. Замирякин, В.П. Часовских // Информация, инновации, инвестиции: Материалы Всероссийской (с международным участием) конференции. – Пермь: Пермский ЦНТИ, 2004. – С. 50-52.
54. Воронов М.П. Применение многоуровневой структуры при создании автоматизированной системы управления организацией в среде СУБД ADABAS / М.П. Воронов, А.С. Каменецких, В.П. Часовских // Известия ОрелГТУ: Материалы II международной научно-технической конференции. - Орел: ОрелГТУ, 2006. - №1 (4). - Т4. - С. 28-31
55. Воронов М.П. Разработка приложений средствами СУБД ADABAS и Natural в распределенной среде в условиях создания информационной системы управления организацией / М.П. Воронов, А.С. Каменецких, Т.А.

Тююшев // Научные труды. – Екатеринбург: Уральский государственный лесотехнический университет, 2006. - Вып. 5. – С. 67-69.

56. Воронов М.П. Построение автоматизированной системы оценки знаний обучающихся в среде ADABAS и Natural / М.П. Воронов, Т.А. Тююшев, В.П. Часовских // XV Международная конференция-выставка “Информационные технологии в образовании”: Сборник трудов участников конференции. – М.: “БИТ про”, 2005. - Часть IV. – С. 29-32.
57. Воронов М.П. Автоматизация деятельности ВУЗа при помощи средств ADABAS и Natural / М.П. Воронов, В.П. Часовских // Компьютерное моделирование 2005: Труды VI международной научно-технической конференции. - СПб.: Изд-во Политехнического университета, 2005. - С. 579-587.
58. Воронов М.П. Моделирование и мониторинг производственно-сбытовых программ лесопромышленных предприятий средствами ADABAS и Natural / М.П. Воронов, В.П. Часовских // Лесной Журнал, 2006. - №1. - С. 119-127.
59. Воронов М.П. Оптимизация хранения и передачи данных в распределенной среде при построении автоматизированной системы управления в среде ADABAS и Natural / М.П. Воронов, В.П. Часовских // Современные наукоемкие технологии, 2005. - №9. - С. 50-51.
60. Воронов М.П. Построение системы комплексной диагностики состояния организации в рамках интегрированной системы управления, внедряемой при помощи средств ADABAS и Natural / М.П. Воронов, В.П. Часовских // Проблемы качества, безопасности и диагностики в условиях информационного общества: Материалы научно-практической конференции. – М.: МИЭМ, 2004. - С. 147-148.
61. Воронов М.П. Система коммуникаций в условиях создания интегрированной системы управления организацией в среде ADABAS и Natural / М.П. Воронов, В.П. Часовских // Научные труды международной научно-практической конференции “СВЯЗЬ-ПРОМ 2005” в рамках 2-го

Евро-Азиатского международного форума “СВЯЗЬПРОМЭКСПО 2005”. - Екатеринбург: ГОУ ВПО УГТУ-УПИ, 2005. - С. 183-188.

62. Воронов М.П. Создание информационных систем управления лесопромышленным предприятием в среде ADABAS и Natural / М.П. Воронов, В.П. Часовских // Лесной Журнал, 2006. - №1. - С. 112-119.

63. Воронов М.П. Управление знаниями как инструмент принятия решений в условиях автоматизированного управления организацией / М.П. Воронов, В.П. Часовских // Конкурентоспособность предприятий и организаций: Сборник статей IV Всероссийской научно-практической конференции. – Пенза: РИО ПСГХА, 2006. - С. 76-79.

64. Воронов М.П. Финансовая информационная модель в рамках построения системы принятия решений в среде ADABAS и Natural / М.П. Воронов, В.П. Часовских // Информационно-вычислительные технологии и приложения: сборник статей IV российско-украинского научно-технического и методического симпозиума. – Пенза: РИО ПГСХА, 2006. - С. 43-46.

65. Геловани В.Л., Башлыков А.А., Бритков В.Б., Вязилов Е.Д. Интеллектуальные системы поддержки принятия решений в нештатных ситуациях с использованием информации о состоянии природной среды. - М.: Эдиториал УРСС, 2001. – 304 с.

66. Глушков В.М. Основы безбумажной информатики. – М.: Наука, 1987. – 552 с.

67. Глушков В.М. Вопросы оптимизации в АСУ. - Вопросы радиоэлектроники. Сер. АСУ, вып. 3, 1980.

68. Глушков В.М. Кибернетика, вычислительная техника, информатика. Избр. тр.: В 3т. / АН УССР, Ин-т кибернетики им В.М. Глушкова. Т.1: Математические вопросы кибернетики. - Киев: Наук. думка, 1990. - 216с.

69. Гордеев А.В., Молчанов А.Ю. Системное программное обеспечение. - СПб.: Питер, 2001.

70. Дик В.В. Методология формирования решений в экономических системах и инструментальные среды их поддержки. – М.: Финансы и статистика, 2001. – 300 с.
71. Жигарев В.А., Игнатов В.П. Метод накопления проектных знаний в экспертных системах. // Тр. ин-та. / ЦНИИпроект, АН СССР. – 1986. - №16. – с. 42-52.
72. Зайцев Н.Г. Информационное и математическое обеспечение АСУП. – Киев: Техника, 1974. – 144 с.
73. Информационные технологии в бизнесе / Под ред. Желены. – СПб: Питер, 2002. – 1120 с.
74. Кастеллани К. Автоматизация решения задач управления: Пер. с франц. – М.: Мир, 1982. – 472 с.
75. Костяков С. ISO 9000 и проблемы информатизации предприятий // PC Week/RE.-1999.-№22.
76. Кроув Т., Эйвисон Д. Базы данных в административных информационных системах: Пер. с англ. / Под ред. О.М. Вейнерова. – М.: Финансы и статистика, 1983. – 168 с.
77. Лаврухин Д.И., Соловьев В.П. Системы и средства автоматизации инженерно-конструкторских работ // Обзор. Системы и технические средства управления. Вып. 16. – М.: ВНТИЦ, 1988. – 96 с.
78. Мамиконов А.Г. Методы разработки автоматизированных систем управления. М., Энергия, 1973.
79. Мидоу Ч. Анализ информационных систем. - М.: Прогресс, 1977. – 400 с.
80. Модин А. А., Яковенко Е. Г. Погребной Е. П. Справочник разработчика АСУ.- 2-е изд., перераб. и доп. – М.: Экономика, 1978. – 583 с.
81. Основы систем автоматизированного проектирования. М. М. Берхеев, А. Н. Козин, И. А. Корниенко, Ю. В. Кожевников. – Издательство Казанского университета, 1988 – 254 с.
82. Овчаров Л.А., Селетков С.Н. Автоматизированные банки данных. – М.: Финансы и статистика, 1982. – 262 с.

83. Основы систем автоматизированного проектирования. / М.М. Берхеев, И.А. Залеев, Ю.В. Кожевников и др. – Казань: Издательство Казанского университета, 1988. – 253 с.
84. Полищук Ю.М., Хон В.Б. Теория автоматизированных банков информации. – М.: Высшая школа, 1989. – 182 с.
85. Свириденко С.С. Современные информационные технологии. – М.: Радио и связь, 1989. – 304 с.
86. Слободин А.В., Часовских В.П. Технология баз данных в системах поддержки принятия решений. // Математические методы и информационные технологии в экономике, социологии и образовании: Сборник статей X международной научно-технической конференции. - Пенза, 2002.
87. Слободин А.В., Часовских В.П. Особенности использования интеллектуальных систем поддержки принятия решений в современных российских условиях. // Научные труды: Сборник. Вып. 2/ Урал. гос. лесотехн. ун-т. - Екатеринбург, 2002. 184 с.
88. Слободин А.В., Часовских В.П. Интеллектуальные системы поддержки принятия решений в управлении организацией. // Научные труды: Сборник. Вып. 2/ Урал. гос. лесотехн. ун-т. - Екатеринбург, 2002. 184 с.
89. Соколова Г.Н. Информационные технологии экономического анализа. – М.: Экзамен, 2002. – 320 с.
90. Справочник проектировщика систем автоматизации управления производством. Под ред. Канд. Техн. Наук Г. Л. Смилянского. Изд. 2-е, перераб. и доп. М. Машиностроение», 1976 – 590 с.
91. Справочник разработчика АСУ. Под ред. Н.П. Федоренко, В.В. Карибского. – М.: Экономика, 1978. – 583 с.
92. Теоретические основы построение автоматизированных систем управления. Разработка технического задания. Воронов А. А. Кондратьев Г. А., Чистяков Ю. В., «Наука», 1977 г. – 232 с.

93. Федоренко Н.П. О разработке научных методов управления народным хозяйством. - "Экономика и математические методы", №3, 1965.
94. Федотов Н.Н., Венчковский Л.Б. Средства информационного обеспечения автоматизированных систем управления. – М.: Издательство стандартов, 1989. – 192 с.
95. Черняк Л. На пути к предприятию, управляемому в реальном времени. // Открытые системы. СУБД. – 2002, №12. с 43-45.
96. Чаудхури С., Дайал У., Ганти В. Технология баз данных в системах поддержки принятия решений. // Открытые системы. СУБД. – 2002, №1. с 37-44.
97. Шихаев К.Н., Пантелеев В.Н., Репьев Ю.М. Процессы интеграции в АСУ. М. Финансы и статистика, 1982.
98. Щиборщ К.В. Интегрированная система управления промышленных предприятий России // Менеджмент в России и за рубежом – 2000, №4.
99. Бритков В. Б. Проблемы поддержки и актуализации данных в информационных системах // Межотраслевая информационная служба, 1997, № 3. С. 41-48.
100. Верников Г. Г. Основные принципы выбора прикладного программного обеспечения для построения корпоративной информационной системы. – М: 2002.
101. Глинских А.А. Мировой рынок КИС // Компьютер-Информ. Май 2000. № 10(80).
102. Гребнев С.А. , Кузякин В .И. , Синенко О. В . Современные подходы к интеграции АСУ // ВКСС Connect!. 2004. №1.
103. Карпачев И.И. Классификация компьютерных систем управления предприятием // PC WEEK/RE, № 44, 1998.
104. Карпачев И.И. О стилях и классах (реальность и мифология компьютерных систем управления предприятием) // PC Week, 2000.
105. Капранов А. КИС: Факторы успешного внедрения http://www.connect.ru/

106. Костяков С. Средства разработки приложений // PC Magazine. 1998. №1. С.169-174.

107. Костяков С. Сила ИТ, власть управления плюс газификация всей Самарской области // Intelligent Enterprise/Корпоративные системы. 2003. №1. С. 31-34

108. Кузнецов С. Д. Проектирование и разработка корпоративных информационных систем. М: Центр Информационных Технологий, 1998.

109. Кузнецов С. Д., Шнитман В.З. Серверы корпоративных баз данных. Информационно-аналитические материалы. М: Центр Информационных Технологий, 1998.

110. Лысенко М.А., Осипов М.Г. Методики анализа и проектирования при построении корпоративных информационных систем // Нефть и Капитал. 2001. №7.

111. Столяров М., Трифаленков И. На пути к управляемым информационным системам. Jet Info - информационный бюллетень. 1999. №3 (70).

112. Ханенко В.Н. Информационные системы. – Л.: Машиностроение. Ленингр. отд-ние, 1988. – 126 с.

113. Воронов М.П. Многофакторное моделирование при анализе и прогнозе доходов автотранспортного предприятия / М.П. Воронов, Л.Ю. Помыткина // Экономическая культура в условиях развития рыночной экономики: отечественная практика и опыт сотрудничества: Межвузовский сборник. - Екатеринбург: ГОУ ВПО УГЛТУ-УПИ, 2003. - Вып.6. – С. 43-49.

114. Воронов М.П. Сравнительный анализ методов прогноза продаж с сезонными колебаниями / М.П. Воронов, Л.Ю. Помыткина // Экономическая культура в условиях развития рыночной экономики: отечественная практика и опыт сотрудничества: Межвузовский сборник. - Екатеринбург: ГОУ ВПО УГЛТУ-УПИ, 2003. - Вып.6. – С. 49-60.

115. Алахов Б.В., Смерс Х., Кокеритц Г. Подготовка первичной информации в АСУП деровообрабатывающей промышленности. М.: Лесная промышленность, 1977.
116. Глушков В.М. и др. Обработка информационных массивов в автоматизированных системах управления. Киев, Наукова думка, 1970.
117. Слободин А.В., Часовских В.П. Автоматизированная система поддержки принятия решения технолога. // Социально-экономические и экологические проблемы лесного комплекса: Сб. матер. междунар. науч.-техн. конф / Урал. гос. лесотехн ун-т. - Екатеринбург, 2003. – 356 с
118. Виногоров Г.К. Технология лесозаготовок. М.: Лесн. пром-ть. 1984. 296 с.
119. Петровский В.С. Оптимальная раскряжевка лесоматериалов. - 2-е издание, перераб. и доп. - М.: Лесная промышленность, 1989. - 288 с.
120. Судьев Н.Т. Лесохозяйственный справочник для лесозаготовителя. М.: Лесн. пром-ть. 1976. - 394 с.
121. Технология заготовки сортиментов на лесосеке: Учебное пособие / И.Р. Шегельман., С.Б. Васильев., И.К. Демин. Петрозаводск: Изд-во ПетрГУ, 1999. - 64с.
122. Шегельман И.Р., Скрыпник В.И., Галактионов О.Н. Техническое оснащение современных лесозаготовок. Петрозаводск: Изд-во Профи-Информ, 2005. – 401 с.
123. Виногоров Г.К. Лесосечные работы. М.: Лесн. пром-ть, 1981. 272 с.
124. Матвейко А.П. Справочник мастера лесозаготовок - Москва:Экология,1993-286с
125. Гацкевич В.А. Справочник мастера лесозаготовок - Лесная промышленность: Москва 1971-392с.
126. Желтов Е.М., Маврина Г.А. Основы технологии и организации труда на лесосеки - Лесная промышленность Москва 1977г.-126с.

127. Шелгунов Ю.В., Кутуков Г.М., Ильин Г.П. Машины и оборудование лесозаготовок, лесосплавного и лесного хозяйства. М.: Лесн. пром-ть, 1982. 520 с.

128. Шелгунов Ю.В., Горюнов А.К., Ярцев И.В. Лесоэксплуатация и транспорт леса. М.: Лесн. пром-ть. 1989. 520 с.

129. Воронов М.П. Основные показатели эффективности применения усовершенствованного метода комбинированных индексов для информационной модели АСУП деревообрабатывающей промышленности / М.П. Воронов, А.В. Слободин, В.П. Часовских // Научные труды. – Екатеринбург: Уральский государственный лесотехнический университет, 2004. - Вып. 3. – С. 150-151.

130. Кузнецов С.Д. Методы оптимизации выполнения запросов в реляционных СУБД // Сб. Итоги науки и техники. Вычислительные науки. Т. 1. – М.: ВИНИТИ, 1989.

131. Слободин А.В. Модели и методы повышения эффективности коммуникаций в базах данных АСУП деревообрабатывающей промышленности. Диссертация на соискание ученой степени кандидата технических наук. - Екатеринбург, 2003.

132. Глушаков С.В., Ломотько Д.В. Базы данных: Учебный курс. – Харьков: Фолио; М.: АСТ, 2000. – 504 с.

133. Горев А. Эффективная работа с СУБД. - Минск: Питер, 1997. - 700 с.

134. Иванов Ю.Н. Теория информационных объектов и систем управления базами данных. – М.: Наука, 1988. - 232 с.

135. Информационные системы общего назначения. Аналитический обзор систем управления базами данных / Под ред. Е.Л. Ющенко. – М.: Статистика, 1975. – 472 с.

136. Когаловский М.Р. Энциклопедия технологий баз данных. – М: Финансы и статистика, 2002. – 800 с.

137. Коннолли, Томас, Бегг, Каролин, Страчан, Анна. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 2-е

изд.: Пер. с англ.: Уч. Пос. – М.: Издательский дом «Вильямс», 2000. – 1120 с.

138. Кузнецов С.Д. Введение в системы управления базами данных: Часть 1. // Системы управления базами данных. - 1995. - №1. - С.14-25.

139. Кузнецов С.Д. Введение в СУБД: Часть 2. // Системы управления базами данных. - 1995. - №2. - С.16-27.

140. Кузнецов С. Д. Основы современных баз данных. М: Центр Информационных Технологий, 1998.

141. Мазур М. Качественная теория информации. – М.: Мир, 1974. – 239 с.

142. Мартэн Д. Базы данных. Практические методы: Пер. с франц. / Под ред. А.В. Шилейко. М.: Радио и связь, 1983. – 168 с.

143. Мейер Д. Теория реляционных баз данных. – М.: Мир, 1987. – 608 с.

144. Системы управления базами данных ДИСОД / Е. С. Броневщук, В. И. Бурдаков, Л. И. Гуков и др.; Под общ. рук. В. И. Дракина. – М.; Финансы и статистика, 1987.-263 с.

145. Системы управления базами данных и знаний. / А.Н. Наумов, А.М. Вендров, В.К. Иванов и др.; под ред. А.Н. Наумова. – М.: Финансы и статистика, 1991. – 351 с.

146. Ульман Дж. Основы систем баз данных. – М.: Финансы и статистика, 1983. - 334с.

147. Флетчер Р. Реляционные базы данных. / в Информационные технологии в бизнесе / под ред. М. Желены. – СПб: Питер, 2002. – с. 730-742.

148. Хансэн Г., Хансэн Д.. Базы данных: разработка и управление: Пер. с англ. – М.: ЗАО «Издательство БИНОМ», 1999. – 704 с.

149. Куцык Б.С. Структура данных и управление. – М.: Наука, 1975. – 125 с.

150. Начао М., Катаяма Т., Уэмура С. Структуры и базы данных: Пер. с япон. / Под ред. В.И. Скворцова. – М.: Мир, 1986. – 197 с.

151. Поспелов Д.А. Логико-лингвистические модели в системах управления. - М.: Энергоиздат, 1981. – 232 с.

152. Тиори Т., Фрай Дж. Проектирование структур баз данных: в 2-х кн. : Пер. с англ. - М.: Мир, 1985. Кн.1 - 287с., Кн.2 – 320 с.

153. Трамбле И.С., Соренсон П. Введение в структуры данных: Пер. с англ. / Под ред. А.Е. Костина, В.Ф. Шаногина. – М.: Машиностроение, 1982. – 784 с.

154. Флорес И. Структуры и управление данными. - М.: Финансы и статистика, 1982. – 319 с.

155. Шаймароданов Р.Б. Моделирование и автоматизация проектирования структур баз данных. – М.: Радио и связь, 1984. – 120 с.

156. Часовских В.П. Лингвистический подход к построению формальной модели базы данных. - Депонир. В ВИНИТИ, 1982, № 3941-82.

157. Buchholz W. File Organization and Addressing, IBM Systems Journal, 2, June 1963, p.86-111.

158. Горохов Д., Чернов В. Методы организации хранения данных в СУБД. // Открытые системы. СУБД. – 2003, №3. с 64-67.

159. Лефковиц Д. Структуры информационных массивов оперативных систем. – М.: Энергия, 1973. 208с.

160. Слободин А.В. Повышение эффективности методов организации хранения и поиска информации для одного класса информационных моделей. // Математические методы и информационные технологии в экономике, социологии и образовании: Сборник статей X международной научно-технической конференции. - Пенза, 2002.

161. Слободин А.В. Организация хранения и поиска информации в системах поддержки принятия технологических решений. // Социально-экономические и экологические проблемы лесного комплекса: Сб. матер. междунар. науч.-техн. конф. - Урал. гос. лесотехн ун-т. - Екатеринбург, 2003. – 356 с.

162. Canning R. G. The Data Administration Function, EDP Anal., 10, 11, p.320-336, (Nov. 1972).

163. Ложе И. Информационные системы: методы и средства: Пер. с фр./ Под ред. К.Л. Корфани и Т.В. Молчановой. М.: Мир, 1979. – 632 с.

164. Уэлдон Дж.-Л. Администрирование баз данных: Пер. с англ. – М.: Финансы и статистика, 1984. – 207 с.

165. Ахтырченко К.В., Леонтьев В.В. Распределенные объектные технологии в информационных системах // СУБД, 1997, №5-6.

166. Дубова Н. Интегрированные системы управления распределенной корпорацией // Открытые Системы, 1998, №1.

167. Кононов А., Кузнецов Е. Онтология распределенных прикладных систем. // Открытые системы. СУБД. – 2002, №11. с 22-28.

168. Кузнецов М. Наследование реализации в распределенных объектных системах. // Открытые системы. СУБД. – 2002, №12. с 60-64.

169. Ладыженский Г.М. Распределенные информационные системы и базы данных // Jet Infosystems, CIT Forum, 1996.

170. Пуха Ю. Объектные технологии построения распределенных информационных систем // СУБД, 1997, №3.

171. Воронов М.П. Сравнительный анализ стоимостей совокупного использования различных СУБД / М.П. Воронов, В.П. Часовских // Вестник УГТУ-УПИ: Труды третьей международной научно-практической конференции Регионального Уральского отделения Академии инженерных наук им. А.М. Прохорова. – Екатеринбург: ГОУ ВПО УГТУ-УПИ, 2004. - № 15 (45). - Ч.1. - С. 167-171.

172. Краснощеков П.С. Принципы построения моделей. - М.: МГУ, 1988. - 264с.

173. Кузнецов Н.А., Кульба В.В., Ковалевский С.С., Косяченко С.А., Методы анализа и синтеза модульных информационно-управляющих систем. – М.: ФИЗМАТЛИТ, 2002. – 800 с.

174. Ладенко И.С. Логические методы построения математических моделей. - Новосибирск: Наука, Сиб отд-ие, 1980. - 192с.

175. Лебедев А.Н. Моделирование в научно-технических системах. - М.: Радио и связь, 1989. - 224с.
176. Математические модели и оптимизация вычислительных алгоритмов: Сб. тр. фак. вычисл. математики и кибернетики МГУ / Под ред. А.Н. Тихонова, А.А.Самарского. - М.: МГУ, 1993. - 254с.
177. Хорошевский В.Ф. Инструментальные экспертные системы // Представление знаний в человеко-машинных и робототехнических информационных систем и среде СУБД. Международная научн. конф. "Программное обеспечение ЭВМ": Тез. докл. – Тверь, 1990. - №4 – с. 12-13.
178. Цаленко М.Ш. Моделирование семантики в базах данных. – М.: Наука, 1989. – 288 с.
179. Шальгин А.С. Прикладные методы статистического моделирования. - Ленинград: Машиностроение, Ленингр. отд-ие, 1986. - 319с.
180. Яглом И.М. Математические структуры и математическое моделирование. - М.: Советское радио, 1980. - 145с.
181. ADABAS: описание утилит. Руководство пользователей.
182. NATURAL: операторы и синтаксис языка. Руководство пользователей.
183. DeBlasis J.P., Johnson T.H. Database Administration - Classical Pattern, Some Experience and Trends, Proc. AFIPS NCC, Vol. 46, AFIPS Press, Arlington, VA, p. 125-137, 1977.
184. Абрамов С.А. Математическое построение и программирование. - М.: Наука, 1978. -191с.
185. Бойко В.В., Савинков В.М. Проектирование баз данных информационных систем. – М.: Финансы и статистика, 1989. – 361 с.
186. Братцева Е.В. Алгоритмы агрегирования. - М. ВЦ АН СССР, 1990. - 56с.
187. Вирт Н. Алгоритмы и структуры данных. – М.: Мир, 1989. – 360 с.

188. Глушков В.М. и др. Математическая информационная среда и проектирование систем искусственного интеллекта - М.: Высшая школа, 1980. –15 с.

189. Глушков В.М., Цейтлин Г.Е., Юшенко Е.Л. Алгебра, языки программирования. - Киев: Наук. думка, 1989. - 376с.

190. Демьяненко В.Ю. Программные средства создания и ведения баз данных. – М.: Финансы и статистика, 1984. – 127 с.

191. Диго С.М. Программирование баз данных. – М.: Финансы и статистика, 1988. – 216 с.

192. Замулин А.В. Системы программирования баз данных и знаний. - Новосибирск: Наука, Сиб. отд-ние, 1990. – 352 с.

193. Калиниченко Л.А. Методы и средства интеграции неоднородных баз данных. – М.: Наука, 1983. – 424 с.

194. Лэнгсам И., Огенстайн М., Тененбаум А. Структуры данных для персональных ЭВМ. – М.: Мир, 1989. – 568 с.

195. Мак-Лоун Р.Р., Крэггс Дж. У. Математическое моделирование. - М.: Мир, 1977. - 277с.

196. Мартин Дж. Организация баз данных в вычислительных системах. – М.: Мир, 1980. - 662с.

197. Бессонов А.Б. Экономическая оценка совокупной стоимости владения информационно-программными системами в организации. / А.Б. Бессонов, А.В. Слободин, В.П. Часовских // Научные труды: - Екатеринбург: Урал. гос. лесотехн. ун-т., 2002. - Вып. 2. С. 184.

198. Воронов М.П. Расчет стоимости совокупного использования СУБД / М.П. Воронов, В.П. Часовских // Вестник УГТУ-УПИ: – Екатеринбург: ГОУ ВПО УГТУ-УПИ, 2004. - № 15 (45). - Ч.1. - С. 171-173.

199. Глинских А.А. Анализ современного состояния мирового рынка КИС // Компьютер-Информ, 2000. - № 11. С. 11-19.

200. Глинских А.А., Андронова О.О. Анализ современного состояния российского рынка КИС // Компьютер-Информ, 2000. - № 12. С. 16-21.

Автоматизированные информационные технологии в экономике: Учебник/М. И. Семенов, И. Т. Трубилин, В. И. Лойко, Т. П. Барановская; Под общ. ред. Трубилина. – М.: Финансы и статистика, 2000. – 416 с.

201. Гордеев А.В., Молчанов А.Ю. Системное программное обеспечение. - СПб.: Питер, 2001.

202. Гэри Хансэн, Джеймс Хансэн. Базы данных: разработка и управление: Пер. с англ. – М.: ЗАО «Издательство БИНОМ», 1999. – 704 с.

203. Евдокимов В. В. И др. Экономическая информатика. Учебник для вузов Под ред. д. э. н. проф. В. В. Евдокимова. – СПб.: Питер, 1997. – 592 с.

204. Информационное и математическое обеспечение АСУП. Зайцев Н. Г. Издание 2-е, исправленное и дополненное. «Техника», 1974, - 144 с.

205. К. Кастеллани. Автоматизация решения задач управления. Пер. с франц. – М.; Мир, 1982.-472 с.

206. Когаловский М. Р. Энциклопедия технологий баз данных. – М.: Финансы и статистика, 2002. – 800 с.

207. Коннолли, Томас, Бегг, Каролин, Страчан, Анна. Базы данных: проектирование, реализация и сопровождение. Теория и практика, 2-е изд.: Пер. с англ.: Уч. Пос. – М.: Издательский дом «Вильямс», 2000. – 1120 с.

208. Кузнецов Н. А., Кульба В. В., Ковалевский С. С., Косяченко С. А. Метод анализа и синтеза модульных информационно-управляющих систем. – М.; ФИЗМАТЛИТ, 2002. – 800 с.

209. Модин А. А., Яковенко Е. Г. Погребной Е. П. Справочник разработчика АСУ.- 2-е изд., перераб. и доп. – М.: Экономика, 1978. – 583 с.

210. Основы систем автоматизированного проектирования. М. М. Берхеев, А. Н. Козин, И. А. Корниенко, Ю. В. Кожевников. – Издательство Казанского университета, 1988 – 254 с.

211. Пустоваров В.И. Ассемблер: программирование и анализ корректности машинных программ: - К.: Издательская группа BHV, 2000

212. Системы управления базами данных ДИСОД. Е. С. Броневщук, В. И. Бурдаков, Л. И. Гуков и др.; Под общ. рук. В. И. Дракина. – М.; Финансы и статистика, 1987.-263 с.

213. Системы управления базами данных и знаний: Справ. Изд./ С40 А. Н. Наумов, А. М. Вендров, В. К. Иванов и др. ; Под ред. А. Н Наумова. – М.: Финансы и статистика, 1991 – 352 с.

214. Справочник проектировщика систем автоматизации управления производством. Под ред. Канд. Техн. Наук Г. Л. Смилянского. Изд. 2-е, перераб. и доп. М. Машиностроение», 1976 – 590 с.

215. Теоретические основы построение автоматизированных систем управления. Разработка технического задания. Воронов А. А. Кондратьев Г. А., Чистяков Ю. В., «Наука», 1977 г. – 232 с.

216. Федотов Н. Н., Венчковский Л. Б. Средства информационного обеспечения автоматизированных систем управления. – М.: Издательство стандартов, 1989. – 192 с.

217. Экономическая информатика / под ред. П. В. Конюховского и Д. Н. Колесова. - СПб Питер, 2000. – 560 с.

218. T.B. Pedersen, C.S. Jensen, C.E. Dyreson, «A Foundation for Capturing and Querying Complex Multidimensional Data», Information Systems, vol. 26, no. 5, 2001

219. P. Vassiliadis, T.K. Sellis, «A Survey of Logical Models for OLAP Databases», ACM SIGMOD Record, vol. 28, no. 4, 1999

220. J. Gray et al., «Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab and Sub-Totals», Data Mining and Knowledge Discovery, vol. 1, no. 1, 1997

221. A. Eisenberg, J. Melton, «SQL Standardization: The Next Steps», ACM SIGMOD Record, vol. 29, no. 1, 2000

222. R. Kimball, The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses, John Wiley & Sons, New York, 1996

223. E. Thomsen, OLAP Solutions: Building Multidimensional Information Systems, John Wiley & Sons, New York, 1997

224. H.-J. Lenz, A. Shoshani, «Summarizability in OLAP and Statistical Data Bases», Proc. 9th Int'l Conf. Scientific and Statistical Database Management, IEEE CS Press, Los Alamitos, Calif., 1997

225. T. Zurek, M. Sinnwell, «Data Warehousing Has More Colours Than Just Black and White», Proc. 25th Int'l Conf. Very Large Databases, Morgan Kaufmann, San Mateo, Calif., 1999

226. UK National Health Service, Read Codes Version 3, Sept. 1999, www.cams.co.uk/readcode.htm (current Nov. 2001)

227. T.B. Pedersen, C.S. Jensen, C.E. Dyreson, «Extending Practical Pre-Aggregation in On-Line Analytical Processing», Proc. 25th Int'l Conf. Very Large Databases, Morgan Kaufmann, San Mateo, Calif., 1999

Михаил Петрович Воронов
Елена Викторовна Кох
Виктор Петрович Часовских
Евгения Васильевна Анянова

КОМПЬЮТЕРНЫЕ МЕТОДЫ ОБРАБОТКИ ИНФОРМАЦИИ В МЕНЕДЖМЕНТЕ ЛЕСОПРОМЫШЛЕННОГО ПРЕДПРИЯТИЯ

Учебное пособие

Компьютерная верстка **М.П. Воронова**

Уральский государственный лесотехнический университет
620100, Екатеринбург, ул. Сибирский тракт, 37

303