

**Федеральное агентство по образованию  
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ГОУВП «АмГУ»**

**УТВЕРЖДАЮ**

**Зав. кафедрой АППиЭ**

\_\_\_\_\_ **А. Н. Рыбалев**

«\_\_\_» \_\_\_\_\_ **2007 г**

**ИНТЕГРИРОВАННЫЕ СИСТЕМЫ ПРОЕКТИРОВАНИЯ И  
УПРАВЛЕНИЯ**

для специальности 22.03.01 – «Автоматизация технологических процессов и  
производств».

**Составитель: ассистент Русинов В. Л.**

**Благовещенск  
2007 г.**

Печатается по решению  
редакционно-издательского  
совета энергетического  
факультета Амурского  
государственного университета.

В. Л. Русинов

Учебно-методический комплекс по дисциплине «Интегрированные системы проектирования и управления» для студентов очной формы обучения для специальности 22.03.01 «Автоматизация технологических процессов и производств».

Учебно-методические рекомендации ориентированы на оказание помощи студентам очной формы обучения по специальности 22.03.01 – «Автоматизация технологических процессов и производств» для формирования теоретических и практических знаний при изучении курса «Интегрированные системы проектирования и управления», выполнении лабораторных работ и расчётно-графической работы.

Амурский государственный университет, 2007.

## СОДЕРЖАНИЕ:

1. Рабочая программа дисциплины.	4
2. График самостоятельной учебной работы студентов по дисциплине.	26
3. Методические рекомендации по самостоятельной работе студентов.	29
4. Методические рекомендации по проведению лабораторных работ.	30
5. План-конспект лекций.	31
6. Методические указания по выполнению лабораторных работ.	250
7. Перечень программных продуктов.	331
8. Комплекты заданий для лабораторных работ.	332
9. Фонд тестовых и контрольных заданий для оценки качества знаний по дисциплине.	333
10. Вопросы к экзамену и зачету.	345
11. Карта обеспеченности дисциплины кадрами профессорско-преподавательского состава.	346

Федеральное агентство по образованию Российской Федерации  
Амурский государственный университет

УТВЕРЖДАЮ  
Проректор по учебной работе  
Е.С. Астапова  
личная подпись, И.О.Ф

«\_\_» \_\_\_\_\_ 200\_\_ г.

РАБОЧАЯ ПРОГРАММА

по дисциплине «**Интегрированные системы проектирования и управления**»

для специальности 22.03.01 «**Автоматизация технологических процессов и производств**», специализаций «Автоматизация технологических процессов тепловых электрических станций», «Автоматизация технико-экономических процессов»

Курс **4, 5** \_\_\_\_\_ Семестр **8, 9** \_\_\_\_\_

Лекции **29** \_\_\_\_\_ (час.) Экзамен **8** \_\_\_\_\_

Практические (семинарские) занятия \_\_\_\_ (час.) Зачет **9** \_\_\_\_\_

Лабораторные занятия **43** \_\_\_\_\_ (час.)

Самостоятельная работа **78** \_\_\_\_\_ (час.)

Всего часов **150** \_\_\_\_\_

Составитель В.Л. Русинов, ассистент кафедры автоматизации  
производственных процессов и электротехники  
(И.О.Ф., должность, ученое звание)

Факультет Энергетический \_\_\_\_\_

Кафедра автоматизации производственных процессов и электротехники \_\_\_\_\_

2007 г.

Рабочая программа составлена на основании Государственного образовательного стандарта ВПО 657900 «Автоматизированные технологии и производства» и учебного плана специальности 22.03.01 «Автоматизация технологических процессов и производств»: блок специальных дисциплин, СД Ф.05 «Интегрированные системы проектирования и управления».

Рабочая программа обсуждена на заседании кафедры автоматизации производственных процессов и электротехники

«\_\_» \_\_\_\_\_ 200\_\_ г., протокол № \_\_\_\_\_

Заведующий кафедрой \_\_\_\_\_ А.Н. Рыбалев

Рабочая программа одобрена на заседании УМС 22.03.01 «Автоматизация технологических процессов и производств»

«\_\_» \_\_\_\_\_ 200\_\_ г., протокол № \_\_\_\_\_

Председатель \_\_\_\_\_ А.Н. Рыбалев

СОГЛАСОВАНО

Начальник УМУ

\_\_\_\_\_ Г.Н. Торопчина

(подпись, И.О.Ф.)

«\_\_» \_\_\_\_\_ 200\_\_ г.

СОГЛАСОВАНО

Председатель УМС факультета

\_\_\_\_\_ Ю.В. Мясоедов

(подпись, И.О.Ф.)

«\_\_» \_\_\_\_\_ 200\_\_ г.

СОГЛАСОВАНО

Заведующий выпускающей кафедрой

\_\_\_\_\_ А.Н. Рыбалев

(подпись, И.О.Ф.)

«\_\_» \_\_\_\_\_ 200\_\_ г.

## ЦЕЛИ И ЗАДАЧИ ИЗУЧЕНИЯ ДИСЦИПЛИНЫ, ЕЕ МЕСТО В УЧЕБНОМ ПРОЦЕССЕ

**Цель преподавания дисциплины «Интегрированные системы проектирования и управления»:**

- обучить студентов методам и основам проектирования и моделирования сложных технических, информационных и экономических систем управления;
- дать основные понятия интегрированной системы, функций и структуры интегрированных систем управления;
- раскрыть взаимосвязь процессов проектирования и математического, методического и организационного обеспечения;
- познакомить с особенностями программно-технических средств для построения моделей интегрированных систем управления, используемых при проектировании автоматизированных систем управления;
- заложить основы визуализационных механизмов системы моделирования для прикладного сопровождения дипломных проектов по специальности 220301 "Автоматизация технологических процессов и производств".

**Задачи изучения дисциплины:**

- освоение методов проектирования и исследования интегрированных систем проектирования и управления;
- практическое освоение студентами современных программных и аппаратных средств проектирования и управления сложных технических и технологических объектов;
- выполнение лабораторного практикума с использованием SCADA-системы Trace Mode-5 и LabView.

**Перечень дисциплин, усвоение которых студентами необходимо при изучении данной дисциплины:**

- «Технические средства автоматизации» – разделы «Измерительные преобразователи» и «Исполнительные механизмы»;
- «Автоматизация технологических процессов и производств» – разделы «Программируемые контроллеры» и «Модульные промышленные компьютеры»;
- «Теория автоматического управления» – разделы «Синтез регуляторов» и «Структуры систем автоматического управления».

Знания и умения, полученные при изучении дисциплины, будут использованы при дипломном проектировании и в практической деятельности выпускника.

## СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

### 1. Федеральный компонент

Государственный образовательный стандарт ВПО 657900 «Автоматизированные технологии и производства» СД.05, Интегрированные системы проектирования и управления:

Интегрированные системы проектирования и управления производствами отрасли: основные понятия интегрированной системы, функции и структуры интегрированных систем, взаимосвязь процессов проектирования, подготовки производства и управления производством, математическое, методическое и организационное обеспечение, программно-технические средства для построения интегрированных систем проектирования и управления; SCADA системы, их функции и использование для проектирования автоматизированных систем управления, документирования, контроля и управления сложными производствами отрасли; примеры применяемых в отрасли SCADA систем.

### 2. Лекционный материал (29 часов)

#### 8 семестр (14 часов)

- 2.1. Введение в дисциплину – 2 часа.
  1. Предпосылки развития современных ИСУ.
  2. Знакомство с Trace Mode 5.
- 2.2. Основные характеристики управляющих систем – 2 часа.
  1. Эффективность. Продуктивность. Доступность.
  2. Кросс-платформенность и дисциплинарность. Открытость ИСУ.
  3. Проект АСУ в Trace Mode 5.
- 2.3. Основные задачи управления ИСУ – 2 часа.
  1. Проблемы развёртывания.
  2. Администрирование пользователей и ресурсов.
  3. Канал Trace Mode 5 (значения канала, обработка данных в канале, классификация каналов).
- 2.4. Современные SCADA-системы – 2 часа.
  1. Введение. Основные возможности и характеристики современных SCADA-систем.
  2. Компоненты систем контроля и управления и их назначение.
  3. Атрибуты канала
    - имя и индекс канала;
    - период и фаза работы канала;
    - время изменения;
    - атрибуты СПАД, регистратор, отчёт;
    - состояние канала;
    - начальное значение канала;
    - флаги архивирования.
- 2.5. Задачи SCADA-систем – 2 часа.
  1. Управление аппаратурой, визуализация, мониторинг

## 2. Атрибуты канала

- интервалы и границы;
- тенденция изменения значения канала;
- достоверность значения канала;
- размерность;
- атрибуты первичной обработки;
- кодировка сигналов;
- комментарий;
- события;
- подключения.

### 2.6. Этапы жизненного цикла промышленных изделий – 2 часа.

1. Этапы жизненного цикла сложных промышленных изделий в концепции CALS технологий.

2. Проектирование.

3. Языки программирования алгоритмов в Trace Mode 5

- язык функциональных блоков (Техно FBD);
- язык инструкций (Техно IL).

### 2.7. Этапы жизненного цикла промышленных изделий – 2 часа.

1. Инженерный анализ

2. Прототипирование

3. Технологическая подготовка производства

4. Производство

5. Разработка графического интерфейса в Trace Mode 5.

- редактор представления данных;
- элементы графического интерфейса.

## 9 семестр (15 часов)

### 2.8. Единое информационное пространство поддержки жизненного цикла изделия – 2 часа.

1. Стандарты

2. Управление данными в автоматизированных системах

3. Архивирование в Trace Mode 5

- общие положения;
- СПАД – локальный архив.

### 2.9. Системы управления производственной информацией – 2 часа.

1. Общие характеристики производителей PDM

2. Функциональные возможности PDM

3. Пользовательская среда

4. Операционная среда

5. Оценки перспективности различных систем PDM

6. Архивирование в Trace Mode 5

- отчёт тревог;
- глобальный регистратор.

### 2.10. Организация взаимодействия с промышленными контроллерами – 2 часа.



1. Организация взаимодействия с промышленными контроллерами
  2. Аппаратная реализация связи с устройствами ввода-вывода
  3. Особенности построения коммуникационного ПО
  4. Реальное время в SCADA – системах
  5. Связь с УСО, контроллерами и приложениями в Trace Mode 5
    - общие сведения;
    - связь с УСО;
    - связь с контроллерами;
    - связь с приложениями (обмен DDE).
- 2.11. Системы PLM – 2 часа.
1. Назначение PLM систем.
  2. Производители PLM систем.
  3. Потребители PLM систем.
  4. Работа в реальном времени в Trace Mode 5
    - запуск MPB;
    - запуск графической консоли;
    - запуск сервера матообработки;
    - настройка DCOM;
    - структура обработки данных.
- 2.12 Надежность SCADA – систем – 2 часа.
1. Архитектура клиент-сервер
  2. Дублирование сервера ввода-вывода
  3. Резервирование на уровне задач
  4. Выделенный сервер файлов
  5. Резервирование сети
  6. Резервирование связи с контроллерами
  7. Разработка распределенных систем
    - идеология распределенных комплексов;
    - уровни АСУ;
    - линии передачи данных;
    - сетевой обмен.
- 2.13 Internet/intranet-решения и SCADA-системы. Стратегия клиентских приложений – 2 часа.
1. Структура Windows DNA;
  2. Новая реализация клиентского приложения в режиме сервер/терминал;
  3. Стратегия клиентских приложений от Wonderware;
  4. Internet/Intranet решения от CiTechnologies;
  5. Общие тенденции и различие реализаций;
  6. Управление через Интернет в Trace Mode 5.
    - консоль управления Web-активатора;
    - создание Web-страниц на сервере;
    - доступ к проекту через Интернет.
- 2.14. Базы данных в ИСУ – 2 часа.

1. Критерии оценки БД
2. Клиент-серверные технологии
3. Базы данных в промышленной автоматизации
4. Базы данных реального времени
5. Документирование в Trace Mode 5
  - организация документирования;
  - редактор шаблонов;
  - сервер документирования.

### **3. Лабораторные работы (43 часа)**

Лабораторные работы выполняются на базе следующих программных средств:

- Trace Mode 5, Adastrа Research Group, Ltd., Россия (учебная версия);
- LabVIEW v.6, National Instruments, USA.

8 семестр (15 часов)

#### **3.1. Знакомство со SCADA-системой Trace Mode. – 3 часа.**

Основные характеристики, состав и назначение отдельных модулей, методика применения. Подходы к проектированию. Установка и настройка Trace Mode 5. Инсталляция инструментальной системы. Общие настройки. Интегрированная среда разработки как инструмент создания единого проекта. Способы разработки. Создание проекта в Trace Mode. Структура проекта. Компоненты проекта: описатель, узел, канал, атрибут. Каналы-вызовы экранов, программ, запросов к БД, документов. Аргументы. Операции в Навигаторе проекта.

#### **3.2. Работа в редакторе базы каналов – 2 часа.**

Создание проекта. Создание узла проекта. Автопостроение базы каналов для контроллера. Редактирование базы каналов. Тиражирование узлов проекта. Автопостроение базы каналов операторской станции для обмена данными с другими узлами проекта. Автопостроение базы каналов операторской станции для обмена данными с внешними контроллерами.

#### **3.3. Разработка и отладка программ. – 2 часа.**

Создание и разработка FBD-программы. Подключение FBD-программы к каналам. Отладка FBD-программы. Создание, разработка и подключение к системе ПЛ-программы.

#### **3.4. Приемы разработки графического интерфейса операторских станций. – 2 часа.**

Создание графической базы узла. Создание статического рисунка. Отображение в графическом виде значений каналов. Тиражирование графических фрагментов. Эмуляция работы графической базы.

#### **3.5. Работа с локальным архивом СПАД и отчетом тревог. – 2 часа.**

Настройка каналов для архивирования. Настройка параметров СПАД. Настройка параметров отчета тревог. Визуализация архивных данных.

3.6. Организация документирования технологических параметров проекта. – 2 часа.

Разработка шаблона. Создание сценария и генерация документа.

3.7. Организация управления технологическим процессом через Интернет. – 2 часа.

Конфигурация сервера HTTP. Создание сайта проекта на сервере. Управление техпроцессом через Интернет.

#### 9 семестр (28 часов)

3.8. Знакомство со средой LabVIEW и разработка виртуального инструмента, включающего два цифровых регулятора и один цифровой индикатор, позволяющего выполнять сложение и умножение двух чисел. – 2 часа.

3.9. Разработка виртуального инструмента для измерения температуры и уровня жидкости в аппарате. Изменение параметров имитировать с помощью датчика случайных чисел. Предусмотреть как графическое отображение параметров, так и цифровое. # датчик случайных чисел – Functions–Numeric–Random number (0–1). – 2 часа.

3.10 Разработка виртуального инструмента для измерения температуры жидкости. # имитация датчика температуры – Functions–Tutorial–Digital Thermometer; # логическое отрицание – Functions–Boolean–Not; # временная синхронизация – Functions–Time&Dialog–Wait Until Next ms Multiple. – 2 часа.

3.11. Построение графика функции  $y = ax^2 + bx + ab$  в диапазоне от -10 до 20 с шагом 2. Предусмотреть возможность изменения параметров  $a$  и  $b$ . – 2 часа.

3.12. Разработка виртуального инструмента для управления наполнением емкости жидкостью при помощи насоса. Предусмотреть возможность регулирования расхода жидкости, сигнализацию заполнения емкости (достижение уровня – 0.95) и автоматическое отключение насоса. – 2 часа.

3.13. Разработка виртуального инструмента, в котором осуществляется наполнение емкости насосом (аналогично работе 3.12), затем 5 секундная задержка на физико-химические превращения (горит сигнальный индикатор) и далее опорожнение емкости. # временная задержка – Functions–Time&Dialog–Wait Until Next ms Multiple. # создание локальной переменной – контекстное меню на индикаторе или регуляторе Create–Local Variable (в нее можно писать или из нее читать – контекстное меню на ней Change To Read Local / Change To Write Local). – 2 часа.

3.14. Разработка виртуального инструмента, отображающего значения функций  $\sin$  и  $\cos$  на одном Waveform Graph за 20 проходов цикла For. – 2 часа.

3.15. Связать и затем развязать кластер, включающий строку (ФИО), массив числовой (оценки экзаменам за сессию), лампочку (горит – переведен на другой семестр, не горит – не переведен). – 2 часа.

3.16. Записать в текстовый файл строку из 10 символов (название функции), 20 числовых значений (значения функции), текущее время, единицу

и ноль (для отображения данных значений на двух лампочках). Затем считать данные значения из файла и отобразить их на соответствующих приборах. – 3 часа.

3.17. Разработать виртуальный инструмент для поддержки CGI интерфейса, позволяющий осуществлять дистанционный запуск vi на другом компьютере, его закрытие, изменение значения его числового (увеличение/уменьшение значения переменной на ) и булевого параметров. Продемонстрировать его работу на базе созданной Internet страницы. – 3 часа.

3.18. Используя DAQ Signal Generator разработать виртуальный инструмент, позволяющий регистрировать значения датчика температуры, включение/ выключение и регулировку интенсивности накала светодиода. – 3 часа.

3.19. Разработать виртуальные инструменты, позволяющие осуществлять передачу данных (включение/выключение тумблера и изменение значения цифрового регулятора) от одного к другому с использованием протокола TCP. – 3 часа.

#### **4. Самостоятельная работа (78 часов)**

Самостоятельная работа по курсу предусматривает:

4.1 Подготовку к лабораторным работам, написание рефератов по теме пройденных лекций осуществляемых с помощью материалов (электронных версий справочных систем программных продуктов, инструкций по эксплуатации промышленных контроллеров и т.д.), предоставленных студентам преподавателем. – 58 часов.

СР 1. [Знакомство с Trace Mode 5.](#)

СР 2. [Редактор базы каналов Trace Mode 5.](#) Справочная система Trace Mode 5, раздел: «Редактор базы каналов», главы: «Общие положения», «Организация редактора базы каналов», «Редактирование проекта», Операции с узлами.

СР 3. [Редактор базы каналов Trace Mode 5.](#) Справочная система Trace Mode 5, раздел: «Редактор базы каналов», глава: «Автопостроение базы каналов».

СР 4. [Редактор базы каналов Trace Mode 5.](#) Справочная система Trace Mode 5, раздел: «Редактор базы каналов», глава: «Редактирование базы каналов», «Объект базы каналов», «Редактирование каналов объекта», «Первичная и выходная обработка», «Отладка алгоритмов».

СР 5. [Языки разработки алгоритмов Trace Mode 5.](#) Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Общие положения», «Язык функциональных блоков»:

- создание и атрибуты FBD-программ;
- вызов программ;
- основные понятия;
- редактирование программ;

- описание функциональных блоков (логические функции, арифметические функции, тригонометрические функции, алгебраические функции, функции сравнения).

CP 6. [Языки разработки алгоритмов Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»:

- описание функциональных блоков (функции выбора, триггеры и счётчики, генераторы).

CP 7. [Языки разработки алгоритмов Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»:

- описание функциональных блоков (управление, ввод/вывод, переходы, обобразование, LD-функции, регулирование, пересылки).

CP 8. [Языки разработки алгоритмов Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»:

- программирование блоков на Техно IL;
- создание блока из FBD-программы;
- подключение внешних алгоритмов.

глава «Язык инструкций».

CP 9. [Архивирование Trace Mode](#). Справочная система Trace Mode 5, раздел: «Архивирование».

CP 10. [Связь с УСО, контроллерами и приложениями Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Связь с УСО, контроллерами и приложениями».

CP 11. [Работа в реальном времени Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Работа в реальном времени».

CP 12. [Разработка распределённых систем Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Разработка распределённых систем», главы: «Обзор раздела», «Идеология распределённых комплексов», «Сетевой обмен по протоколу NetBEUI», «Обмен по протоколу M-Link».

CP 13. [Разработка распределённых систем Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Разработка распределённых систем», главы: «Обмен по коммутируемым линиям», «Обмен по GSM», «Управление через интернет».

CP 14. [Документирование Trace Mode 5](#). Справочная система Trace Mode 4.2. Выполнение расчетно-графической работы «Разработка ИСУ технологическим процессом в Trace Mode». – 20 часов.

Технологический процесс и система управления либо предлагаются студентом самостоятельно по результатам прохождения производственной практики и в порядке подготовке к курсовому проектированию по дисциплине «Автоматизация технологических процессов и производств», либо назначаются преподавателем. Контроль выполнения – проверка РГР.

## 5. Перечень и темы промежуточных форм контроля знаний

Промежуточный контроль знаний студентов по дисциплине предусматривает две контрольные точки в 8 семестре и одну в 9 семестре, оценки по которым выставляются на основе информации о выполнении лабораторных работ и РГР, а также на основе тестирования теоретических знаний, полученных за прошедший период обучения. Предусмотрено тестирование по темам:

5.1. Принципы структурной организации интегрированной системы управления техническими средствами, анализ сигналов в интегрированных системах управления техническими средствами. – 1 конт. точка, 8 семестр.

5.2. Проектирование программного обеспечения интегрированных систем управления. Объектно-ориентированный подход к проектированию программного обеспечения. – 2 конт. точка, 8 семестр.

5.3. Программно-технические средства построения интегрированных систем проектирования и управления. Организация взаимодействия с промышленными контроллерами. – 1 конт. точка, 9 семестр.

## **6. Итоговый контроль знаний**

### 6.1. Общие положения.

Итоговый контроль знаний по дисциплине предусматривается в формах экзамена (8 семестр) и зачета (9 семестр).

### 6.2. Экзамен.

#### 6.2.1. Вопросы к экзамену:

1. Предпосылки и основные характеристики развития современных ИСУ.
2. Основные функции интегрированных систем управления.
3. Этапы жизненного цикла промышленных изделий.
4. Понятия «жесткого» и «мягкого» реального времени.
5. Взаимосвязь процессов проектирования, подготовки производства и управления производством.
6. Основные принципы структурной организации интегрированной системы управления техническими средствами.
7. Общая характеристика ИСУ с сетевой архитектурой.
8. Типовые микропроцессорные устройства обработки информации.
9. Основные функциональные задачи управления техническими средствами.
10. Детерминированные сигналы в интегрированных системах управления.
11. Случайные процессы в интегрированных системах управления.
12. Назначение, структура и состав систем информационной поддержки.
13. Информационные потоки в системах информационной поддержки.
14. Оценка характеристик вычислительных элементов СИП.
15. Модельное обеспечение СИП. Разработка модельного обеспечения СИП. Модели данных и знаний.
16. Понятие метода и технологии проектирования ПО.

17. Сущность структурного подхода к проектированию программного обеспечения.
18. Метод функционального моделирования SADT. Моделирование потоков данных (процессов).
19. Сравнительный анализ SADT- моделей и диаграмм потоков данных.
20. Сущность объектно-ориентированного подхода к проектированию программного обеспечения.
21. Общая характеристика унифицированного языка моделирования UML.
22. Диаграммы классов, взаимодействия, состояний в UML.
23. Диаграммы деятельности, компонентов, размещения в UML.
24. Сопоставление и взаимосвязь структурного и объектно-ориентированного подходов.

#### 6.2.2. Критерии допуска и оценки.

Для допуска к экзамену достаточными основаниями являются выполнение, сдача, проверка и защита всех лабораторных работ. Защита работы предполагает опрос студента по теме работы и ее результатам. В порядке исключения к экзамену может также быть допущен студент, не сдавший одну или две лабораторные работы.

Экзамен предусматривает 2 теоретических вопроса. Студент, не сдавший одну или две лабораторные работы и допущенный к экзамену в порядке исключения, отвечает также дополнительные вопросы по темам данных работ. Для подготовки ответа студенту отводится 40 мин. Для получения удовлетворительной оценки достаточно показать знание основных понятий по теме вопроса. Оценка «хорошо» выставляется студенту, показавшему способность экономического, математического, технического и др. обоснований применяемых решений и подходов по автоматизации установки, процесса и т.д. Оценка «отлично» выставляется, если, кроме того, студент правильно ответил на дополнительные вопросы по темам, смежным с темами основных вопросов. При этом неправильные ответы на дополнительные вопросы могут служить основанием для снижения оценки до «удовлетворительно», если эти ответы свидетельствуют о слабом понимании материала.

#### 6.3. Зачет (9 семестр).

##### 6.3.1. Вопросы к зачету:

1. Предпосылки и причины появления CALS-технологий.
2. Этапы жизненного цикла промышленных изделий.
3. Автоматизация технологической подготовки производства.
4. Автоматизация проектирования электронных устройств.
5. Автоматизация производственных и логистических процессов.
6. Математическое обеспечение CALS-технологий.

7. Сущность задачи координации в интегрированных автоматизированных системах управления производством.
  8. Общая схема иерархического управления в ИАСУ.
  9. Математическая постановка задачи межуровневой координации.
- Применение алгоритмов координации в промышленных системах управления.
10. Назначение, функции и использование SCADA-систем для проектирования автоматизированных систем управления, документирования, контроля и управления сложными производствами.
  11. Компоненты систем контроля и управления в SCADA-системах.
  12. Обобщённая двухуровневая схема реализации АСКИУ.
  13. Основные задачи SCADA-систем.
  14. Организация взаимодействия SCADA-систем с промышленными контроллерами.
  15. Механизмы обмена данными DDE и OLE.
  16. Особенности построения коммуникационного ПО. Реальное время в SCADA - системах.
  17. Назначение и состав MES/EAM/HRM-система реального времени
  18. Основные характеристики T-Factory 6 .

### 6.3.2. Критерии допуска и оценки.

Для допуска к зачету достаточными основаниями являются выполнение, сдача, проверка и защита всех лабораторных работ и РГР. Защита работы предполагает опрос студента по теме работы и ее результатам. В порядке исключения к зачету может также быть допущен студент, не сдавший одну или две лабораторные работы.

Зачет предусматривает 2 теоретических вопроса. Студент, не сдавший одну или две лабораторные работы и допущенный к зачету в порядке исключения, отвечает также дополнительные вопросы по темам данных работ. Для подготовки ответа студенту отводится 20 мин. Для получения зачета достаточно показать знание основных понятий по теме вопроса.

## УЧЕБНО-МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ

### 1. Основная литература

- 1.1. Н.П. Деменков. SCADA-системы как инструмент проектирования АСУ ТП. Учебное пособие. Москва. 2004. Изд. МГТУ им. Н.Э. Баумана, 326 с .
- 1.2. Матвейкин В.Г., Фролов С.В., Шехтман М.Б.- Применение SCADA-систем при автоматизации технологических процессов.
- 1.3. Амбросовский В.М., Белый О.В., Скороходов Д.А. Интегрированные системы управления технических средств транспорта / Под ред. Ю.А.Лукомского – СПб.: «Элмор», 2001.

### 2. Дополнительная литература

- 2.1. Буч Г., Рамбо Д., Джексопсон А Язык UML. Руководство пользователя. Пер с англ – М.: ДМК, 2000. - 432 с.



2.2. Шемелин В.К. Проектирование систем управления в машиностроении. Учебник для вузов. – М.: изд-во «Станкин», 1998. - 254 с.

### **3. Программные руководства**

3.1. Справочная система ТРЕЙС МОУД Adastra Research Group, Ltd.,  
Россия

3.2. LabVIEW v.6, National Instruments, USA.

## УЧЕБНО-МЕТОДИЧЕСКАЯ (ТЕХНОЛОГИЧЕСКАЯ) КАРТА ДИСЦИПЛИНЫ

Номер недели	Номер темы	Вопросы, изучаемые на лекции	Занятия (номера)		Используемые наглядные и методические пособия	Самостоятельная работа студентов		Формы контроля
			практич. (семина.)	лаборат.		содержание	час.	
1	2	3	4	5	6	7	8	9
<b>8 семестр</b>								
1	1	Введение в дисциплину. 1. Предпосылки развития современных ИСУ. 2. Знакомство с Trace Mode 5.				Подготовка к выполнению лабораторной работы.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1, экзамен
2				№1		Подготовка отчета по лабораторной работе.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1, экзамен
3	2	Основные характеристики управляющих систем. 1. Эффективность. Продуктивность. Доступность. 2. Кросс-платформенность и дисциплинарность. Открытость ИСУ. 3. Проект АСУ в Trace Mode 5.				Подготовка к выполнению лабораторной работы.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1, экзамен
4				№2		Подготовка отчета по лабораторной работе.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1, Экзамен

1	2	3	4	5	6	7	8	9
5	3	<p>Основные задачи управления.</p> <p>1. Проблемы развёртывания.</p> <p>2. Администрирование пользователей и ресурсов.</p> <p>3. Канал Trace Mode 5 (значения канала, обработка данных в канале, классификация каналов).</p>				Подготовка к выполнению лабораторной работы.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1, экзамен
6				№3		Подготовка отчета по лабораторной работе.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №2, Экзамен
7	4	<p>Современные SCADA-системы.</p> <p>1. Введение. Основные возможности и характеристики современных SCADA-систем.</p> <p>2. Компоненты систем контроля и управления и их назначение.</p> <p>3. Атрибуты канала</p> <ul style="list-style-type: none"> <li>- имя и индекс канала;</li> <li>- период и фаза работы канала;</li> <li>- время изменения;</li> <li>- атрибуты СПАД, регистратор, отчёт;</li> <li>- состояние канала;</li> <li>- начальное значение канала;</li> <li>- флаги архивирования.</li> </ul>				Подготовка к выполнению лабораторной работы.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №2, Экзамен
8				№4		Подготовка отчета по лабораторной работе.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №2, Экзамен

1	2	3	4	5	6	7	8	9
9	5	<p>Задачи SCADA-систем.</p> <p>1. Управление аппаратурой, визуализация, мониторинг</p> <p>2. Атрибуты канала</p> <ul style="list-style-type: none"> <li>- интервалы и границы;</li> <li>- тенденция изменения значения канала;</li> <li>- достоверность значения канала;</li> <li>- размерность;</li> <li>- атрибуты первичной обработки;</li> <li>- кодировка сигналов;</li> <li>- комментарий;</li> <li>- события;</li> <li>- подключения.</li> </ul>				Подготовка к выполнению лабораторной работы.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №2, экзамен
10				№5		Подготовка отчета по лабораторной работе.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №2, Экзамен
11	6	<p>Этапы жизненного цикла промышленных изделий.</p> <p>1. Этапы жизненного цикла сложных промышленных изделий в концепции CALS технологий.</p> <p>2. Проектирование.</p> <p>3. Языки программирования алгоритмов в Trace Mode 5</p> <ul style="list-style-type: none"> <li>- язык функциональных блоков (Техно FBD);</li> <li>- язык инструкций (Техно IL).</li> </ul>				Подготовка к выполнению лабораторной работы.	2	Проверка и защита лабораторной работы. экзамен
12				№6		Подготовка отчета по лабораторной работе.	2	Проверка и защита лабораторной работы. Экзамен.

1	2	3	4	5	6	7	8	9
13	7	Этапы жизненного цикла промышленных изделий. 1. Инженерный анализ 2. Прототипирование 3. Технологическая подготовка производства 4. Производство 5. Разработка графического интерфейса в Trace Mode 5. - редактор представления данных; - элементы графического интерфейса.				Подготовка к выполнению лабораторной работы.	2	Проверка и защита лабораторной работы. экзамен
14				№7		Подготовка отчета по лабораторной работе.	2	Проверка и защита лабораторной работы. Экзамен
15							2	Экзамен
<b>9 семестр</b>								
1	8	Единое информационное пространство поддержки жизненного цикла изделия. 1. Стандарты 2. Управление данными в автоматизированных системах 3. Архивирование в Trace Mode 5 - общие положения; - СПАД – локальный архив.		№8		Подготовка к выполнению лабораторной работы и отчета по ней.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1. Зачет
2				№9		Подготовка к выполнению лабораторной работы и отчета по ней.	2	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1. Зачет

1	2	3	4	5	6	7	8	9
3	9	Системы управления производственной информацией. 1. Общие характеристики производителей PDM 2. Функциональные возможности PDM 3. Пользовательская среда 4. Операционная среда 5. Оценки перспективности различных систем PDM 6. Архивирование в Trace Mode 5 - отчёт тревог; - глобальный регистратор.		№ 10		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1. Проверка и защита РГР. Зачет.
4				№11		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1. Проверка и защита РГР. Зачет.
5	10	Организация взаимодействия с промышленными контроллерами. 1. Организация взаимодействия с промышленными контроллерами 2. Аппаратная реализация связи с устройствами ввода-вывода 3. Особенности построения коммуникационного ПО 4. Реальное время в SCADA – системах 5. Связь с УСО, контроллерами и приложениями в Trace Mode 5 - общие сведения; - связь с УСО; - связь с контроллерами;		№12		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Контрольная точка и тестирование №1. Проверка и защита РГР. Зачет.
6				№13		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Проверка и защита РГР. Зачет.

1	2	3	4	5	6	7	8	9
7	11	Системы PLM. 1. Назначение PLM систем. 2. Производители PLM систем. 3. Потребители PLM систем. 4. Работа в реальном времени в Trace Mode 5 - запуск MPB; - запуск графической консоли; - запуск сервера матобработки; - настройка DCOM; - структура обработки данных.		№14		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Проверка и защита РГР. Зачет.
8				№15		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Проверка и защита РГР. Зачет.
9	12	Надежность SCADA – систем. 1. Архитектура клиент-сервер 2. Дублирование сервера ввода-вывода 3. Резервирование на уровне задач 4. Выделенный сервер файлов 5. Резервирование сети 6. Резервирование связи с контроллерами 7. Разработка распределенных систем - идеология распределенных комплексов; - уровни АСУ; - линии передачи данных; - сетевой обмен.		№ 16		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Проверка и защита РГР. Зачет.
10				№17		Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.	4	Проверка и защита лабораторной работы. Проверка и защита РГР. Зачет.





1	2	3	4	5	6	7	8	9
11	13	<p>Internet/intranet-решения и SCADA-системы. Стратегия клиентских приложений.</p> <ol style="list-style-type: none"> <li>1. Структура Windows DNA;</li> <li>2. Новая реализация клиентского приложения в режиме сервер/терминал;</li> <li>3. Стратегия клиентских приложений от Wonderware;</li> <li>4. Internet/Intranet решения от CiTechnologies;</li> <li>5. Общие тенденции и различие реализаций;</li> <li>6. Управление через Интернет в Trace Mode 5.</li> </ol> <ul style="list-style-type: none"> <li>- консоль управления Web-активатора;</li> <li>- создание Web-страниц на сервере;</li> <li>- доступ к проекту через Интернет.</li> </ul>		№18		<p>Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.</p>	4	<p>Проверка и защита лабораторной работы. Проверка и защита РГР. Зачет.</p>
12				№19		<p>Подготовка к выполнению лабораторной работы и отчета по ней. Выполнение РГР.</p>	4	<p>Проверка и защита лабораторной работы. Проверка и защита РГР. Зачет.</p>
13	14	<p>Базы данных в ИСУ.</p> <ol style="list-style-type: none"> <li>1. Критерии оценки БД</li> <li>2. Клиент-серверные технологии</li> <li>3. Базы данных в промышленной автоматизации</li> <li>4. Базы данных данных реального времени</li> <li>5. Документирование в Trace Mode 5</li> </ol> <ul style="list-style-type: none"> <li>- организация документирования;</li> <li>- редактор шаблонов;</li> <li>- <b>сервер документирования.</b></li> </ul>					2	<p>Проверка и защита лабораторной работы. Зачет</p>
14							2	<p>Проверка и защита лабораторной работы. Зачет</p>

**2. ГРАФИК**  
**самостоятельной учебной работы студентов по дисциплине**  
**«Автоматизация технологических процессов».**

№ недели	№ тем ы	Тема	Самостоятельная работа студентов		Формы контроля
			Содержание	Часы	
1	2	3	4	5	6
8 семестр					
1	1	СР 1. <a href="#">Знакомство с Trace Mode 5.</a>	Подготовка к лабораторной работе и составление отчета. Написание реферата.	6	Зачет, сдача лабораторных работ
3	2	СР 2. <a href="#">Редактор базы каналов Trace Mode 5.</a> Справочная система Trace Mode 5, раздел: «Редактор базы каналов», главы: «Общие положения», «Организация редактора базы каналов», «Редактирование проекта», «Операции с узлами».	Подготовка к лабораторной работе и составление отчета. Написание реферата.	6	Зачет, сдача лабораторных работ, проверка и защита реферата.
5	3	СР 3. <a href="#">Редактор базы каналов Trace Mode 5.</a> Справочная система Trace Mode 5, раздел: «Редактор базы каналов», глава: «Автопостроение базы каналов».	Подготовка к лабораторной работе и составление отчета. Написание реферата.	6	Контрольная точка и тестирование №1, зачет, сдача лабораторных работ, проверка и защита реферата
7	4	СР 4. <a href="#">Редактор базы каналов Trace Mode 5.</a> Справочная система Trace Mode 5, раздел: «Редактор базы каналов», глава: «Редактирование базы каналов», «Объект базы каналов», «Редактирование каналов объекта», «Первичная и выходная обработка», «Отладка алгоритмов».	Подготовка к лабораторной работе и составление отчета. Написание реферата.	6	Контрольная точка и тестирование №1, зачет, сдача лабораторных работ, проверка и защита реферата

1	2	3	4	5	6
9	5	<p>СР 5. <a href="#">Языки разработки алгоритмов Trace Mode 5</a>. Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Общие положения», «Язык функциональных блоков»: - создание и атрибуты FBD-программ; - вызов программ; - основные понятия; - редактирование программ; - описание функциональных блоков (логические функции, арифметические функции, тригонометрические функции, алгебраические функции, функции сравнения).</p>	<p>Подготовка к лабораторной работе и составление отчета. Написание реферата.</p>	6	<p>Контрольная точка и тестирование №1, зачет, сдача лабораторных работ, проверка и защита реферата.</p>
11	6	<p>СР 6. <a href="#">Языки разработки алгоритмов Trace Mode 5</a>. Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»: - описание функциональных блоков (функции выбора, триггеры и счётчики, генераторы).</p>	<p>Подготовка к лабораторной работе и составление отчета. Написание реферата.</p>	6	<p>Контрольная точка и тестирование №1, зачет, сдача лабораторных работ.</p>
13	7	<p>СР 7. <a href="#">Языки разработки алгоритмов Trace Mode 5</a>. Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»: - описание функциональных блоков (управление, ввод/вывод, переходы, обобразование, LD-функции, регулирование, пересылки).</p>	<p>Подготовка к лабораторной работе и составление отчета. Написание реферата.</p>	6	<p>Контрольная точка и тестирование №2, зачет, сдача лабораторных работ, проверка и защита реферата.</p>
<b>9 семестр</b>					
1	8	<p>СР 8. <a href="#">Языки разработки алгоритмов Trace Mode 5</a>. Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»: - программирование блоков на Техно IL; - создание блока из FBD-программы; - подключение внешних алгоритмов. глава «Язык инструкций».</p>	<p>Подготовка к лабораторной работе и составление отчета. Написание реферата.</p>	6	<p>Контрольная точка и тестирование №2, зачет, сдача лабораторных работ, проверка и защита реферата.</p>

1	2	3	4	5	6
3	9	СР 9. <a href="#">Архивирование Trace Mode</a> . Справочная система Trace Mode 5, раздел: «Архивирование».	лабораторной работе и составление отчета. Написание реферата.	6	Контрольная точка и тестирование №2, зачет, сдача лабораторных работ, проверка и защита реферата
5	10	СР 10. <a href="#">Связь с УСО, контроллерами и приложениями Trace Mode 5</a> . Справочная система Trace Mode 5, раздел: «Связь с УСО, контроллерами и приложениями».	Подготовка к лабораторной работе и составление отчета. Написание реферата.	6	Контрольная точка и тестирование №2, зачет, сдача лабораторных работ, проверка и защита реферата
7	11	СР 11. <a href="#">Работа в реальном времени Trace Mode 5</a> . Справочная система Trace Mode 5, раздел: «Работа в реальном времени».	Подготовка к лабораторной работе и составление отчета.	6	Контрольная точка и тестирование №2, зачет, сдача лабораторных работ.
9	12	СР 12. <a href="#">Разработка распределённых систем Trace Mode 5</a> . Справочная система Trace Mode 5, раздел: «Разработка распределённых систем», главы: «Обзор раздела», «Идеология распределённых комплексов», «Сетевой обмен по протоколу NetBEUI», «Обмен по протоколу M-Link».	Подготовка к лабораторной работе и составление отчета.	6	Контрольная точка и тестирование №2, зачет, сдача лабораторных работ.
11	13	СР 13. <a href="#">Разработка распределённых систем Trace Mode 5</a> . Справочная система Trace Mode 5, раздел: «Разработка распределённых систем», главы: «Обмен по коммутируемым линиям», «Обмен по GSM», «Управление через интернет».	Подготовка к лабораторной работе и составление отчета.	6	Зачет, сдача лабораторных работ.
13	14	СР 14. <a href="#">Документирование Trace Mode 5</a> . Справочная система Trace Mode	Подготовка к лабораторной работе и составление отчета.	6	Зачет, сдача лабораторных работ. Защита РГР
15					Зачет, сдача лабораторных работ. Защита РГР

### **3. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО САМОСТОЯТЕЛЬНОЙ РАБОТЕ СТУДЕНТОВ.**

Самостоятельная работа студентов по дисциплине предусматривает следующие виды работ:

1. Предварительная подготовка к лабораторным работам и составление отчетов по ним. – 18 часов;
2. Написание рефератов – 40 часов.
3. Работа над РГР – 20 часов.

Тематика рефератов и список рекомендуемой литературы представлены в рабочей программе.

#### **4. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ РАБОТ.**

Лабораторные занятия предусмотрены в рабочей программе в объёме 43 часов. Тематика лабораторных работ и последовательность их выполнения представлены в рабочей программе.

Лабораторные занятия проводятся в учебной аудитории оснащённой персональными компьютерами с установленным на них ПО:

1. Trace Mode 5.
2. LabVIEW 6.

## 5. ПЛАН-КОНСПЕКТ ЛЕКЦИЙ.

### ЛЕКЦИЯ 1 ВВЕДЕНИЕ В ДИСЦИПЛИНУ

1. Предпосылки развития современных интегрированных систем проектирования и управления ИСПУ.
2. Знакомство с Trace Mode 5

#### 1. Предпосылки развития современных ИСУ

В современном бизнесе уже невозможно обойтись без опоры на сложные инфосистемы с распределенной архитектурой. Принято считать, что появление и развитие платформы Unix, ПК, локальных сетей и приложений клиент/сервер стало основным стимулом для развертывания в корпорациях распределенной информационной инфраструктуры. Но нельзя забывать и о том, что сегодня существенно меняются и сами принципы ведения бизнеса. Становится нормой распределение ответственности за принятие решений на большое число независимых бизнес - подразделений корпорации.

Меняются информационные потребности внутри предприятия. В традиционно независимых друг от друга областях - производстве, маркетинге и продажах - совместно используется информация и накопленный опыт. Для оперативного принятия решений руководителям корпорации необходимы средства сравнения результатов деятельности независимых бизнес - модулей. Наконец, организации должны обеспечивать своих клиентов нужной информацией не только в центральном офисе, но и в географически разбросанных филиалах и в режиме мобильного доступа. Попытаемся сформулировать общие принципы интегрированного управления и описать, какие преимущества оно способно дать современному предприятию.

Перестройка организационной структуры предприятий и новые задачи бизнеса увеличивают нагрузку на компьютерные системы. При этом и сама индустрия информационных технологий претерпевает с начала 90-х гг. серьезные изменения. Одна из тенденций еще в 1992 г. была обозначена компанией IDC как "дезинтеграция": многие производители ПК и серверных систем стали отказываться от одновременного выпуска аппаратуры и ОС, и обращаться к другим компаниям. Это относится и к области бизнес-приложений. Вместо того чтобы использовать монолитные прикладные серии, для решения своих задач корпорации стремятся объединять лучшие программные компоненты от различных поставщиков.

В результате дезинтеграции, в большинстве организаций информационная поддержка бизнеса обеспечивается крайне неоднородными и на аппаратном, и на прикладном уровне средами. Теперь даже простой мониторинг и модификация информационных ресурсов на множестве различных систем -

уже весьма обременительная задача для ИС – менеджеров (ИС – инф. системы), а распознавание возникающих затруднений и их причин может стать настоящим кошмаром. В распределенной среде клиент/сервер сложной задачей является правильная конфигурация клиентских настольных систем и различных сетевых компонентов. Приложения клиентских ПК могут существовать в различных версиях для поддержки разных вариантов соответствующих машин. В корпорации постоянно расширяются и обновляются прикладные и аппаратные системы, реорганизовывается сама структура предприятия - и все это должно безотлагательно находить отражение в соответствующей среде информационной поддержки бизнеса.

В этих условиях нелегко приходится системным администраторам, на которых ложится очень большая нагрузка. В их задачи входит развертывание и модификация новых аппаратных и прикладных решений, учет пользователей и предоставление им доступа к необходимым ресурсам, поддержка надежной работы взаимосвязанных друг с другом систем, сетей и приложений. С другой стороны, в современной корпорации эффективное управление такой сложной информационной средой становится одним из главных условий успешного ведения бизнеса. Традиционная практика использования управляющих средств, для различных частей распределенной среды такую эффективность обеспечить не в состоянии в силу ограниченности возможностей и разобщенности этих систем. Администратору большой неоднородной среды необходима интегрированная управляющая система, которая позволит ему развертывать новые продукты по всему предприятию и поддерживать их в рабочем состоянии.

**Интегрированная система управления – это единый программно технический комплекс, который обеспечивает взаимосвязь процессов производства, снабжения, планирования, контроля, реализации на предприятии, и позволяет получать информацию о хозяйственной деятельности предприятия в любой момент времени.**

## **2. Знакомство с Trace Mode 5**

### **Многоязыковая поддержка**

При инсталляции ТРЕЙС МОУД, можно задать язык копируемой справочной системы и файлов readme.

Язык меню и диалогов ТРЕЙС МОУД определяется по идентификатору языковых настроек (LCID) операционной системы. С помощью диалога **Языки и стандарты** панели управления Windows этот параметр можно изменять, однако следует иметь в виду, что некоторые диалоги ТРЕЙС МОУД содержат стандартные сообщения ОС, язык которых с помощью диалога **Языки и стандарты** изменить нельзя.

Вывод всех команд и сообщений, а также ввод необходимых параметров на требуемом языке обеспечивается при запуске ТРЕЙС МОУД под соответствующей локализацией Windows.

ТРЕЙС МОУД поддерживает следующие языки: русский, английский (США), итальянский, китайский (упрощенное письмо).



## Функциональная структура

ТРЕЙС МОУД - это программный комплекс, предназначенный для разработки, настройки и запуска в реальном времени систем управления технологическими процессами. Все программы, входящие в ТРЕЙС МОУД, делятся на две группы:

- инструментальная система разработки АСУ;
- исполнительные модули (runtime).

Рассмотрим, какие программы входят в этих группы, и разберемся с назначением каждой из них.

### Инструментальная система

Инструментальная система включает в себя три редактора:

- Редактор базы каналов;
- Редактор представления данных;
- Редактор шаблонов.

В них разрабатываются: база данных реального времени, программы обработки данных и управления, графические экраны для визуализации состояния технологического процесса и управления им, а так же шаблоны для генерации отчетов о работе производства.

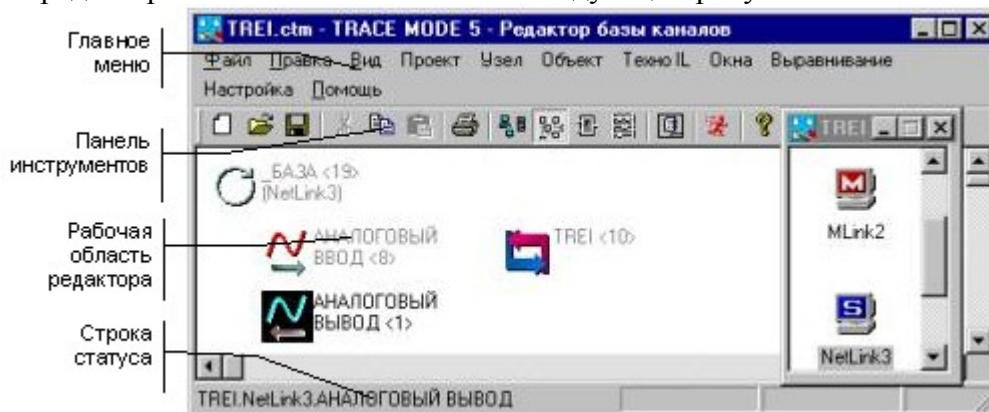
В зависимости от лицензии инструментальная система позволяет создавать проекты на разное количество каналов. Существуют следующие градации инструментальных систем по количеству точек ввода/вывода в одном узле проекта: 128, 1024, 32000x16, 64000x16.

Рассмотрим назначение редакторов инструментальной системы.

### Редактор базы каналов

В **Редакторе базы каналов** создается математическая основа системы управления: описываются конфигурации рабочих станций, контроллеров и УСО, а также настраиваются информационные потоки между ними. Здесь же описываются входные и выходные сигналы и их связь с устройствами сбора данных и управления. В этом редакторе задаются периоды опроса или формирования сигналов, настраиваются законы первичной обработки и управления, технологические границы, программы обработки данных и управления. Здесь настраивается архивирование технологических параметров, сетевой обмен, а также решаются некоторые другие задачи.

Вид окна редактора базы каналов показан на следующем рисунке.



Результатом работы в этом редакторе является математическая и информационная структуры проекта АСУТП. Эти структуры включают в себя набор баз каналов и

файлов конфигурации для всех контроллеров и операторских станций (узлов) проекта, а также файл конфигурации всего проекта.

Файл конфигурации проекта имеет расширение **ctm** и сохраняется в рабочей директории системы разработки. Для хранения всех остальных файлов проекта в рабочей директории создается каталог, имя которого совпадает с именем файла конфигурации. При этом базы каналов сохраняются в файлы с расширениями **dbb**.

Для запуска редактора базы каналов следует выбрать соответствующий ярлык в папке инструментальной системы. Можно также из командной строки запустить модуль **СНВ.EXE**.

### Редактор представления данных

Вход в редактор представления данных осуществляется либо двойным нажатием ЛК на соответствующем ярлыке в папке ТРЕЙС МОУД 5.0, либо запуском исполнительного модуля **picman.exe**.



На приведенном выше рисунке демонстрируется общий вид окна редактора представления данных. Здесь разрабатывается графическая часть проекта системы управления. При этом создается статичный рисунок технологического объекта, а затем поверх него размещаются динамические формы отображения и управления. Среди них такие, как поля вывода численных значений, графики, гистограммы, кнопки, области ввода значений и перехода к другим графическим фрагментам и т. д.

Кроме стандартных форм отображения (ФО), ТРЕЙС МОУД позволяет вставлять в проекты графические формы представления данных или управления, разработанные пользователями. Для этого можно использовать стандартный механизм Active-X.

Все формы отображения информации, управления и анимационные эффекты связываются с информационной структурой, разработанной в редакторе базы каналов.

Графические базы узлов проекта, созданные в редакторе представления данных, сохраняются в файлах с расширением **dbg**. Их сохранение осуществляется в соответствующие директории проектов.

### Редактор шаблонов

Для разработки шаблонов отчетов о хорде технологического процесса в инструментальную систему входит специальный редактор - Редактор шаблонов. Чтобы войти в него, следует дважды нажать ЛК на соответствующем ярлыке папки инструментальной системы. Другой способ - запустить с командной строки **htmpled.exe**.

Организация экрана редактора шаблонов представлена на следующем рисунке.



### *Исполнительные модули (RUNTIME)*

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 1. [Знакомство с Trace Mode 5](#).  
Справочная система Trace Mode 5, раздел: «Начало изучения».**

## ЛЕКЦИЯ 2

# ОСНОВНЫЕ ХАРАКТЕРИСТИКИ УПРАВЛЯЮЩИХ СИСТЕМ

1. Эффективность. Продуктивность. Доступность.
2. Кросс-платформенность и дисциплинарность. Открытость ИСУ.
3. Проект АСУ в Trace Mode 5.

### 1. Эффективность. Продуктивность. Доступность.

Распределенные информационные системы охватывают практически все аспекты работы современного предприятия, делая все более тесной связь между производственными объектами и компонентами информационной инфраструктуры. Руководители компаний тратят на информационные технологии огромные суммы, фактически, с одной лишь целью - повысить свои шансы в конкурентной борьбе. Естественно, им необходимы гарантии надежности работы сети и отдельных систем, доступности жизненно важных для бизнеса приложений, а также общей эффективности использования информационной инфраструктуры. Непродуманная организация управления инфосистемами не может дать таких гарантий. Анализируя имеющиеся сегодня системы управления и их влияние на работу корпорации, можно выделить три параметра их оценки.

**Эффективность** - сколько сетевых устройств, серверов или настольных систем может находиться в ведении одного администратора.

**Продуктивность** - время, необходимое администратору для выполнения действий по поддержке и повышению эффективности работы сети, систем и приложений.

**Доступность** - время, в течение которого пользователи имеют доступ к ресурсам вычислительной среды.

**Эффективность** управляющей системы показывает, насколько хорошо организован труд администраторов. При использовании эффективной системы развитие бизнеса будет опережать процесс разрастания штата специалистов, необходимых для поддержки информационной инфраструктуры предприятия. Такая система управления позволяет одному администратору поддерживать большее количество узлов (серверов, пользователей, сетевых устройств) и выполнять больше операций управления удаленно. Это сокращает число перемещений, которые приходится делать менеджеру, и соответственно снижает связанные с этим затраты.

Практика показывает, что большинство сбоев сетей и систем происходит не по вине нерадивых пользователей из-за ошибок конфигурации, допущенных администраторами информационных систем. Сосредотачивая нити управления в руках меньшего числа менеджеров, эффективная система управления стимулирует выработку согласованных методов изоляции и разрешения проблемных ситуаций. В результате реже возникают ошибки на стадиях начальной конфигурации распределенной среды и обеспечивается более быстрое и качественное восстановление после сбоев.

Эффективная система позволяет обеспечить то, что принято называть проактивным управлением средой. Согласованность методов и широкий охват управляемых систем дает администраторам возможность заранее определять те проблемы, которые могут возникнуть в различных информационных ресурсах. Согласованные и проактивные методы управления в свою очередь позволяют рационально выбирать необходимые технологии. Имея полную информацию о ресурсах, которые поддерживают решение той или иной задачи, менеджеры смогут найти оптимальный вариант использования существующих систем, тогда как администратор с узким кругозором скорее прибегнет к покупке нового оборудования.

**Продуктивность**, также имеет очень важное значение. Производительная система снижает затраты на выполнение ежедневных операций, высвобождая время администратора инфосреды для продуктивного анализа существующих систем, оптимизации их производительности и изоляции потенциальных источников проблем. Таким образом, администратор получает возможность разрабатывать и быстро воплощать в жизнь новые идеи.

Наконец, такая характеристика управляющей системы, как обеспечение **доступности сетевых и системных ресурсов**, является крайне значимой для современного предприятия. Бизнес настолько сильно увязан с распределенной средой корпорации, что цена простоев, может оказаться слишком высока, особенно для пользователей, так или иначе вовлеченных в работу с системами удаленного доступа. В большинстве организаций отсутствие доступа к разделяемым ресурсам, возможностям внутренней и внешней связи и данным в Internet может значительно снизить продуктивность работы и привести к большим потерям в прибыли предприятия.

Доступность ресурсов подразумевает доступность бизнес-приложений. Реальное значение параметров различных сетевых устройств, серверов и настольных систем определяется их способностью обеспечить согласованное и надежное функционирование приложений. Соответственно, качество управляющей системы определяется тем, насколько она способна гарантировать производительность и надежность на уровне приложений, обеспечивающих ведение бизнеса. Поскольку эффективность приложений имеет такое значение для работы предприятия, интеграция данных о ресурсах, трафике и производительности приложений в единой управляющей среде становится ключевым фактором успешного использования информационных технологий.

Чтобы добиться эффективного и продуктивного управления информационной средой с высоким уровнем доступности бизнес-приложений, организации стараются выбирать лучшие средства и системы управления. Однако поиск таких средств может оказаться нелегкой задачей, поскольку эта область сейчас активно развивается и набор возможностей и функций, которые реализуют среды управления, постоянно пополняется (Рис.1.)

Рис. 1.



Системы начала 90-х гг. обеспечивали простой сбор данных об управляемых объектах и по сути в очень малой степени позволяли осуществлять реальное управление. Только вкладывая значительные средства в консультации специалистов со стороны, пользователи таких систем могли извлечь из голых управляющих данных необходимые сведения для эффективной поддержки своих информационных сред.

Большинство современных управляющих систем дает менеджерам представление о состоянии информационной среды и средства контроля над системами. Средства определения сетевой топологии, управление событиями и возможности удаленной конфигурации обеспечивают так называемое "реактивное" управление, ориентированное на быстрое исправление неполадок и эффективность повседневного сопровождения распределенной среды.

Однако для действительно эффективного управления администраторам нужнее анализ использования сети за определенный период времени и тенденций возникновения ошибочных ситуаций, нежели переизбыток статистики производительности в режиме реального времени. Только такой "исторический" анализ сетей, характерный для систем с проактивным подходом к управлению, способен обеспечить поддержку принятия решений. То же самое относится и к управлению клиентскими и серверными системами и распределенными приложениями.

К концу десятилетия будут разработаны управляющие средства с автоматизированным ответом (automated management response), которые сделают реальностью отказоустойчивую информационную среду. Наиболее заманчивой перспективой для поставщиков программного и аппаратного обеспечения является возможность оборудовать свои продукты средствами самовосстановления (self-healing), которые могут сделать их работу максимально эффективной. Однако такая среда потребует беспрецедентной степени интеграции механизмов сбора данных, аналитических средств и систем устранения неисправностей и будет напрямую зависеть от степени зрелости методов корреляции событий, позволяющих скоординировать процесс анализа ошибок и реакцию на них.

## **2. Кросс-платформенность и дисциплинарность. Открытость ИСУ.**

Вложения в интегрированные системы управления чаще всего дают гораздо более высокую отдачу, чем использование отдельных управляющих средств для различных системных и сетевых ресурсов. Что же должна представлять собой действительно интегрированная управляющая среда?

С появлением распределенных систем отдельные приложения управления стали использовать одну и ту же сетевую инфраструктуру. Это дало им возможность, по крайней мере, теоретически, разделять информацию в различных операционных средах. Так возникла интеграция достаточно низкого уровня, которая обеспечивала доступ к каждому управляющему приложению из отдельного "окна", в общем пользовательском интерфейсе. Данная архитектура управления позволяла администратору работать с множеством систем с одной консоли но, по сути, не обеспечивала реальной интеграции, поскольку возможности разделения данных от различных приложений были очень ограниченными.

В действительно интегрированной среде одно управляющее приложение имеет единый интерфейс и единое представление данных для различных вычислительных платформ. С другой стороны, такая среда должна предоставлять администратору возможность

разделять информацию между различными дисциплинами управления. Иными словами, основные характеристики интегрированной системы управления следующие:

- Кросс-платформенность - приложение, которое реализует функции отдельной дисциплины управления, прозрачно для различных операционных сред;
- Кросс-дисциплинарность - приложения для различных дисциплин используют общую информацию;
- Открытость - возможность интеграции средств управления других поставщиков.
- Кросс-дисциплинарные возможности обеспечивают совместную работу различных управляющих модулей и тем самым повышают эффективность всей системы в целом. Например, можно интегрировать средства управления программным обеспечением с приложением управления хранением. В результате программа резервирования будет информирована о том, какие прикладные системы устанавливались в последнее время, и выполнять резервирование только при необходимости.

В полностью интегрированной среде управления должен быть реализован унифицированный, открытый способ просмотра и разделения информации, который может использоваться всеми входящими в эту среду, управляющими приложениями на всех вычислительных платформах. Интегрированная среда должна:

1. обладать согласованным пользовательским интерфейсом (например, интерфейс реального мира CA-Unicenter TNG);
2. иметь возможность разделять информацию между различными операционными средами и дисциплинами управления; реализация этой возможности подразумевает наличие общего, возможно распределенного, репозитория данных и объектно-ориентированной архитектурной базы (например, объектно-ориентированная база интегрированного семейства управляющих приложений TME 10 компании Tivoli);
3. обеспечивать представление информационной инфраструктуры как с точки зрения системного и сетевого управления, так и исходя из интересов бизнеса (например, представление бизнес-процессов в CA-Unicenter TNG);
4. быть распределенной как физически, так и логически;
5. обеспечивать иерархическую организацию управления - возможность делегирования прав менеджера сверху вниз и передачи ответственности за выполнение определенных действий снизу вверх.

В большой, мультиплатформенной, распределенной вычислительной среде ежедневно приходится выполнять множество управляющих "транзакций": генерацию сообщений о событиях, модификацию учетной информации пользователя, распределение нового программного обеспечения, операции по управлению хранением данных, сбор информации о производительности и т.д. Использование интегрированной системы управления, удовлетворяющей этим условиям, может существенно повысить эффективность работы. Интеграция позволяет администратору за одну операцию охватить множество платформ одновременно и предотвратить возникновение ошибок из-за повторения однотипных действий. Автоматическая корреляция событий с разных платформ также повышает качество работы менеджера.

Открытость управляющей среды реализуется с помощью прикладных программных интерфейсов и других средств. Эти возможности позволяют интегрировать новые продукты, а также те системы, которые уже использовались в организации и по-прежнему представляют ценность для нее, тем самым, сохраняя инвестиции.

### 3. Проект АСУ в Trace Mode 5

ТРЕЙС МОУД - это программный пакет для разработки проектов автоматизации любой сложности. Это могут быть как небольшие технологические установки (10-15 контролируемых параметров), так и крупные объекты, распределенные по большой территории, реализующие контроль и управление десятками тысяч параметров.

Проект ТРЕЙС МОУД включает в себя программное обеспечение (ПО) для всех используемых АРМ и контроллеров – узлов проекта. Математические и графические компоненты их программного обеспечения одновременно загружаются в редакторы. Это делает проект прозрачным для разработчика и облегчает настройку взаимодействия узлов проекта и обмена данными.

Основными понятиями, относящимися к структуре разработки систем управления в ТРЕЙС МОУД, являются:

Проект;

Узел;

Объект;

Автопостроение.

#### ***Проект***

Начнем с определения проекта в ТРЕЙС МОУД.

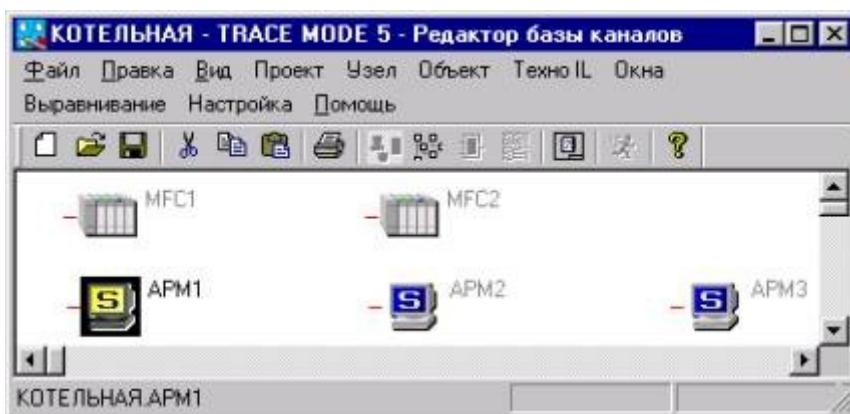
Проект – это совокупность всех математических и графических компонентов ПО для операторских станций и контроллеров одной АСУТП, объединенных информационными связями и единой системой архивирования.

Проект может содержать сотни узлов, а может включать в себя только один контроллер или операторскую станцию.

#### **Структура проекта**

Структура проекта описывается и редактируется в редакторе базы каналов и сохраняется в файле конфигурации проекта. Он имеет расширение **ctm** и записывается в директорию ТРЕЙС МОУД. Все компоненты проекта хранятся в отдельных файлах в поддиректории с тем же именем, что у файла конфигурации проекта.





После загрузки проекта в редактор базы каналов на экран выводится окно редактирования структуры проекта. В рабочем поле этого окна выводятся изображения узлов, входящих в проект (рабочие станции и контроллеры). Чтобы перейти к редактированию базы каналов любого узла проекта, следует дважды нажать ЛК на его графическом идентификаторе.

Проект ТРЕЙС МОУД включает в себя программное обеспечение всех входящих в него узлов, которые могут быть связаны между собой по локальной сети, по последовательным интерфейсам, по коммутируемым линиям или по радиоканалу. Проект размещается на каждом узле, при этом наименование поддиректорий его размещения на разных узлах должно быть одним и тем же. Не используемое на узле программное обеспечение может быть удалено из проекта, размещенного на этом узле (например, базы каналов и графические базы других узлов).



Графическая часть - это совокупность всех экранов для представления данных и супервизорного управления, входящих в графические базы узлов проекта.

Создание и редактирование графической составляющей проекта осуществляется в **редакторе представления данных**. Структура проекта представлена здесь в виде дерева, корневыми элементами которого являются имена узлов, а вложенными – имена групп и экранов. Она выводится в **Навигаторе проекта** в бланке **Экраны**. После выбора нужного экрана он выводится в рабочее поле для редактирования.

## Узел

Проект ТРЕЙС МОУД включает в себя программное обеспечение всех входящих в него операторских станций и контроллеров.

Любое устройство, в котором запущено программное обеспечение ТРЕЙС МОУД, реализующее серверные функции называется **узлом**. Это может быть контроллер, операторская или архивная станция.

Узлы одного проекта могут быть связаны между собой по локальной сети, по последовательным интерфейсам, по коммутируемым линиям или по радиоканалу. При описании разработки систем управления по отношению к узлам будут применяться два термина:

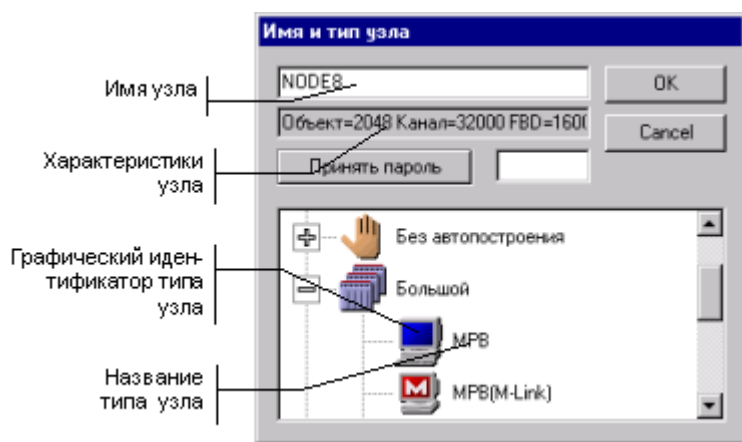
текущий узел;

удаленный узел.

Первый из них – это узел, рассматриваемый в данный момент. Второй – это тот, с которым обменивается данными текущий узел.

## Создание узла

Чтобы создать новый узел, надо выполнить команду **Создать** из меню **Узлы**. При этом на экране появится следующий диалог.



Здесь для создаваемого узла надо указать тип и задать имя (до 20 символов без пробелов). Тип узла определяет, какой монитор будет использован для его запуска, а также механизмы автопостроения его базы каналов.

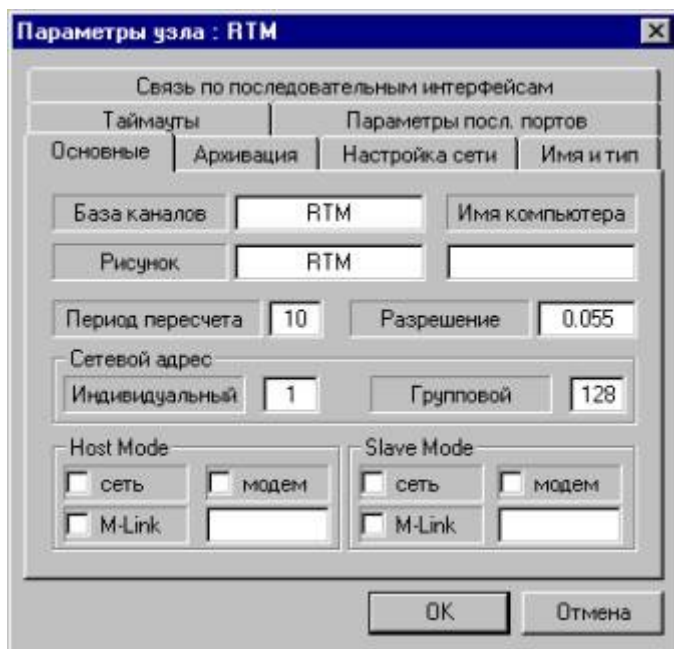
Все узлы разбиты по четырем классам. Класс узла определяет его информационную мощность и уровень системы управления, на котором он будет использоваться.

После создания нового узла в рабочем поле редактора появляется его изображение, включающее в себя его имя и графический идентификатор, соответствующий установленному типу.



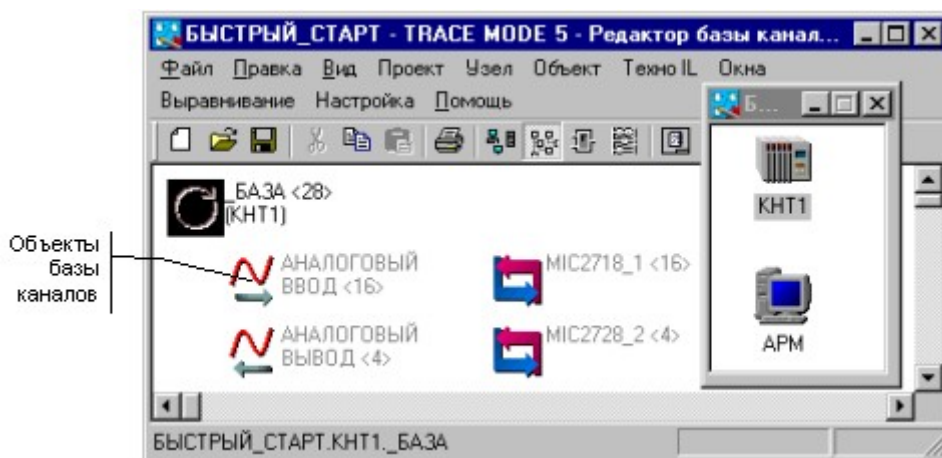
### Редактирование и настройка узла

Для настройки параметров узла или редактирования его базы каналов надо выделить его в окне структуры проекта. Это осуществляется нажатием ЛК на его графическом идентификаторе. Чтобы настроить параметры выделенного узла, надо нажать ПК на его изображении, а для перехода к редактированию базы каналов - ЛК. Примерный вид диалога настройки параметров узла показан на следующем рисунке.



**Внимание!** Имена базы каналов и рисунка не должны содержать пробелов.

При переходе к редактированию базы каналов запускается процедура автопостроения (ее описание приведено ниже). После настройки новых связей для автопостроения осуществляется вход в окно редактирования узла, показанный на следующем рисунке.



Здесь изображаются объекты базы каналов текущего узла.

## Объекты базы каналов

Для каждого узла проекта создается база данных реального времени. В ТРЕЙС МОУД она называется базой каналов и имеет иерархическую структуру. Основным элементом базы является **канал**. О каналах речь будет идти ниже. Каналы одной базы могут группироваться по заданным признакам или произвольно. Оформленные группы каналов могут быть подчинены друг другу и создавать таким образом иерархические структуры. Такие группы называются **объектами базы каналов**.

Объект базы каналов - это группа каналов, которой приписан набор атрибутов. Над каналами объектов могут осуществляться групповые операции.

### Настройка параметров объекта

#### Графический идентификатор

#### Имя объекта

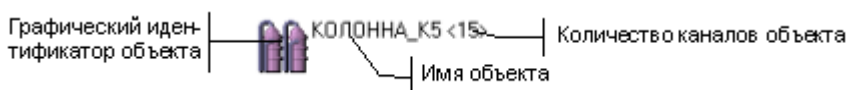
Объект базы каналов имеет следующий набор параметров: имя, графический идентификатор, подчиненность, загружаемость и состояние его каналов при старте. Они настраиваются в диалоге **Параметры объекта**. Его вид представлен на следующем рисунке.



### Графический идентификатор

Для графической идентификации объекта можно использовать одну из стандартных иконок. Это позволяет легко ориентироваться в больших базах каналов.

Следующий рисунок демонстрирует изображение объекта в редакторе базы каналов.



### Имя объекта

Имя объекта представляет собой текстовую строку длиной до 20 любых символов. Имя используется для идентификации объекта при ссылках на содержащиеся в нем каналы.

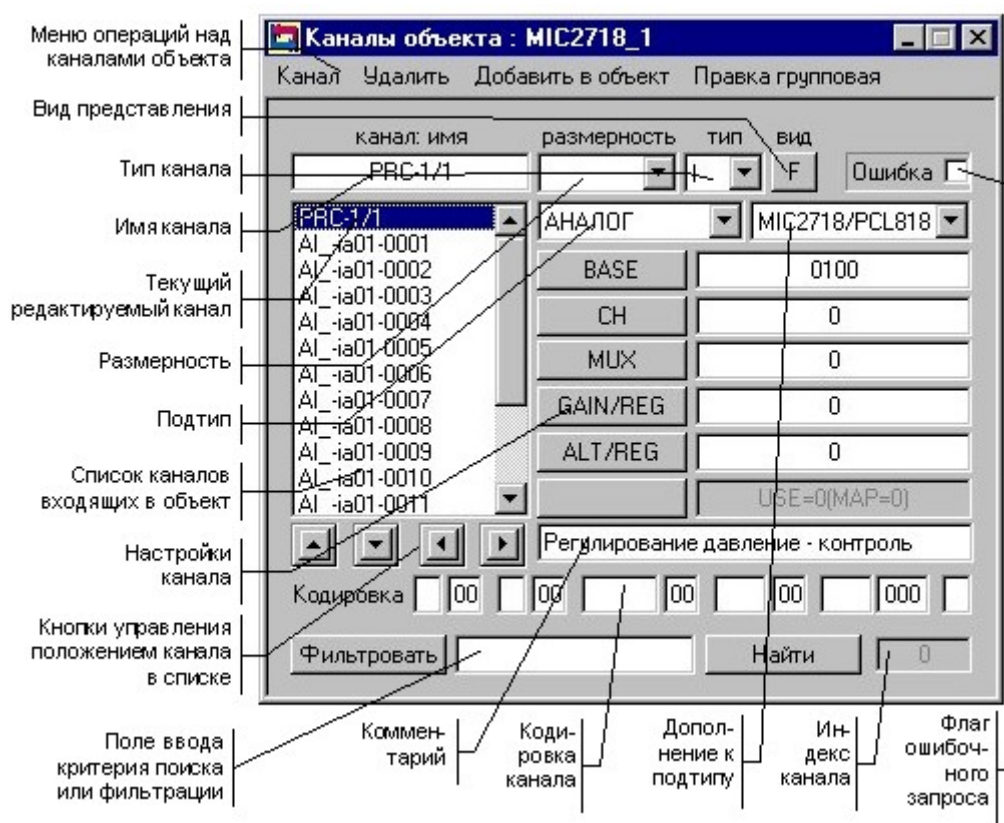
### Стандартные объекты

При создании узла в его базе каналов формируется набор стандартных объектов. Они заполняются каналами по заданным признакам (например, каналы, запрашивающие данные по сети). Для них заранее определены графические идентификаторы и имена.

Количество и типы стандартных объектов зависят от класса узла. Например, объекты - посадочные места плат УСО, создаются только для узлов класса **Контроллеры**.

В базе каналов обязательно присутствует объект **БАЗА**. В него автоматически добавляются все каналы, создаваемые в текущей базе.

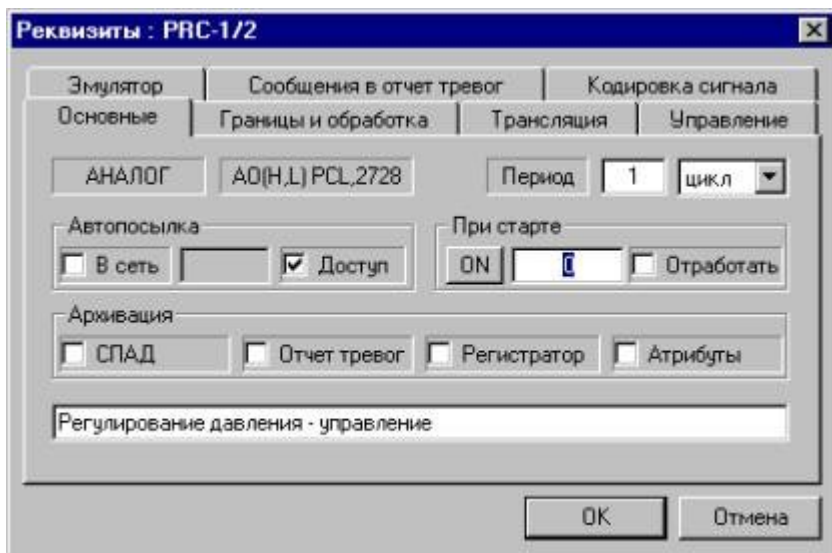
### Редактирование каналов объекта



Двойное нажатие ЛК на изображении объекта в окне редактирования базы каналов выводит на экран диалог **Каналы объекта**. В нем можно редактировать, добавлять и удалять каналы из текущего объекта. Вид этого диалога показан на следующем рисунке.

Выбор любого канала из списка приводит к выводу на экран диалога **Реквизиты**, в котором настраиваются все атрибуты выбранного канала. Вид диалога представлен на следующем рисунке.



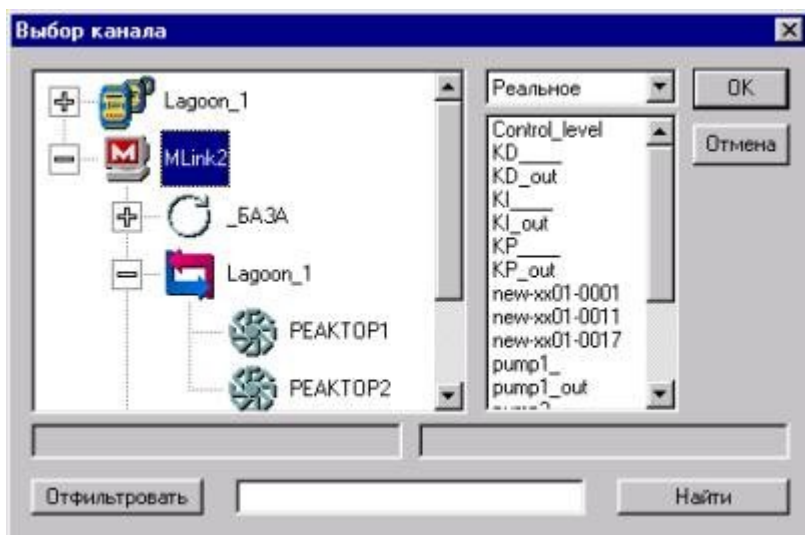


Подробное описание этих диалогов приведено в разделе, посвященном описанию канала ТРЕЙС МОУД.

### Подчиненность объектов

Объекты базы каналов можно собирать в произвольные структуры. При этом каждый объект может входить в группу и быть подчиненным по отношению к объекту-родителю группы и одновременно быть родителем другой группы.

Такая группировка используется для структурирования проекта. Она не влияет на пересчета каналов и объектов.



Более подробно подчиненность объектов рассматривается в разделе, посвященном редактору базы каналов.

### Автопостроение

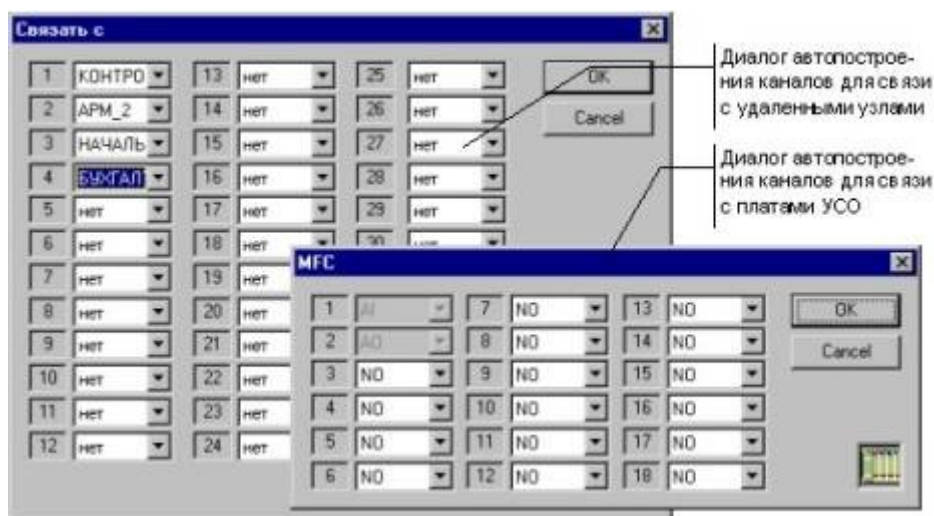
В рамках создания проекта автоматизации необходимо описать информационные потоки: для контроллеров надо создать и настроить каналы обмена данными с платами УСО, а для АРМ – каналы обмена данными с контроллерами и другими узлами проекта.

В крупных проектах задача заполнения баз, описывающих адреса источников и приемников данных, является очень трудоемкой. Это может привести к ошибкам, обусловленным объемом и рутинностью данной работы.

Для облегчения этой работы в ТРЕЙС МОУД реализованы следующие шесть механизмов автопостроения баз каналов:

- создание каналов обмена данными с удаленными узлами текущего проекта;
- создание каналов обмена данными с платами УСО;
- создание каналов обмена данными с контроллерами нижнего уровня (не РС-совместимыми);
- создание каналов при импорте баз данных;
- создание каналов обмена с OPC-серверами;
- создание каналов обмена данными с каналами объекта удаленного узла.

На рисунке показаны диалоги настройки автопостроения для посадочных мест контроллера и для связи с удаленными узлами.



Подробнее автопостроение будет рассматриваться в разделах, посвященных описанию редактора базы каналов, обмена данными с УСО и создания распределенных систем управления.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 2. [Редактор базы каналов Trace Mode 5](#).  
Справочная система Trace Mode 5, раздел: «Редактор базы каналов», главы: «Общие положения», «Организация редактора базы каналов», «Редактирование проекта», «Операции с узлами».**

## ЛЕКЦИЯ 3. ОСНОВНЫЕ ЗАДАЧИ УПРАВЛЕНИЯ

1. Проблемы развёртывания.
2. Администрирование пользователей и ресурсов.
3. Канал Trace Mode 5:
  - значения канала;
  - обработка данных в канале;
  - классификация каналов.

Как показали исследования IDC, использование автоматизированных управляющих систем, и в особенности интегрированных сред управления, существенно повышает продуктивность решения следующих основных задач:

- развертывание системного и прикладного обеспечения (инсталляция и расширение);
- администрирование пользователей и ресурсов (защита и контроль доступа);
- поддержка доступности информационных технологий (обнаружение и коррекция ошибок и общее сопровождение).

### **1 Проблемы развертывания**

Если в эпоху мэйнфреймов установка новых систем и приложений была достаточно простой операцией, то с появлением ПК и распределенных систем клиент/сервер ситуация значительно усложнилась. На это повлияли такие факторы, как дезинтеграция аппаратного и программного обеспечения, сложность распределенных приложений, ужесточение пользовательских требований к гибкости системы и необходимость быстро реагировать на изменения в организациях. В информационной среде корпораций аппаратные расширения бывают как минимум раз в год, реинжиниринг основного ПО необходим каждые несколько лет, а незначительные модификации в прикладной области происходят постоянно. В результате возникает необходимость в автоматизации управления распределением и модификациями различных типов. В качестве примеров можно назвать расширения базовых операционных сред (например, миграция от WinNT к Unix), модификации базовых офисных приложений (например, Office 2000), модификации различных важных утилит, например, антивирусных систем, модификации драйверов и т.д. Дополнительные проблемы порождают клиентские компьютеры, которые часто требуют тщательной индивидуальной настройки.

По данным опроса IDC, ИТ-менеджеры тратят в среднем 190 часов в месяц на процесс развертывания систем для 100 пользователей. Половина этого времени уходит на инсталляцию и расширение прикладного ПО.



Естественно, менеджеры заинтересованы в повышении эффективности этих операций с помощью автоматизированных средств управления развертыванием. Сегодня, основные решения в этой области включают: диски быстрого старта (quickstart disk), которые используются для ускорения начальной установки клиентских и серверных машин; приложения автоматизированного распознавания аппаратного и программного обеспечения, которые позволяют осуществлять быстрый начальный учет ресурсов и автоматизируют модификации; продукты электронного распределения ПО, реализующие передачу файлов по локальным и глобальным сетям на множество настольных компьютеров и серверов.

К инструментальному средству или набору средств для управления развертыванием предъявляются различные технические требования, однако все их можно свести к двум базовым принципам - масштабируемости и управляемости.

**Масштабируемость** - характеризует производительность управляющей системы: ее возможности работать с большим числом различных конфигураций, распределять большие файлы, выполнять размещение по тысячам или десяткам тысяч настольных систем и по большому числу различных соединений в локальных и глобальных сетях.

**Управляемость** означает возможность управления развертыванием без вмешательства человека. Кроме того, эффективное средство управления развертыванием должно иметь возможность реализовывать управление на основе политики, выработанной самой организацией (так называемый метод policy-based). Такое управление будет определяться описаниями конкретных заданий, подразделений, функциональных ролей на предприятии и т.д.

Использование интегрированных сред управления в процессах развертывания позволяет почти вдвое сократить затрачиваемое на эти задачи время. Представим себе, например, администратора, который решает задачу распределения ПО в клиент/серверной системе отдела корпорации. При этом клиентские компоненты приложения должны выполняться на ПК с Windows 95 и Unix, а серверные - на машинах с NT и Unix. Управляющая система с простой поддержкой множества вычислительных платформ будет передавать приложение с Windows на Windows и с Unix на Unix, и только кросс-платформенное решение способно распределять ПО из единого хранилища на все необходимые системы одновременно с помощью одного управляющего приложения, учитывая при этом конкретные требования данной платформы. Примером такого управляющего приложения может служить Tivoli TME 10 Courier, которое, действуя вместе с модулем управления ресурсами TME 10 Inventory, позволяет развертывать приложения по всем разнородным компонентам информационной системы предприятия, от центров данных на базе мэйнфреймов до Web-серверов, и гарантирует не только корректную инсталляцию ПО, но и его правильное функционирование.

## 2 Администрирование пользователей и ресурсов

Как и в случае с развертыванием систем, модель клиент/сервер усложняет процесс администрирования пользователей, охватывающий теперь и все операции по управлению доступом к корпоративным ресурсам. К таким ресурсам относятся персональные учетные данные пользователей и файлы в их личных каталогах, прикладные системы корпоративного масштаба, например, электронная почта, стандартные и специфические приложения, системные ресурсы: факс-серверы, Internet и т.п.

Системы, которые обычно предоставляют доступ к таким ресурсам - это сетевые машины, мобильные компьютеры и любые из многочисленных серверов и хостов организации. Таким образом, в распределенной среде ответственность за управление ресурсами "распыляется" по множеству неоднородных систем. И это порождает массу проблем, среди которых наиболее значительная - несогласованность методов выполнения таких административных операций, как добавление пользователей, создание паролей, определение правил и прав доступа к приложениям и данным.

Кроме того, некоторые функции и возможности могут быть реализованы на одной системе, и отсутствовать на другой. Например, в большинстве Unix-систем пользовательские пароли доступны администратору, их можно свободно просмотреть и отредактировать. А NetWare не разрешает ни одному пользователю (даже с правами супервизора) увидеть текущие пароли. Быстрые изменения информационной инфраструктуры современной корпорации усложняют не только управление развертыванием систем, но и процессы администрирования. Расширения операционных сред клиентских компьютеров, добавление новых систем и общая реорганизация корпорации требуют, чтобы права доступа быстро приводились в соответствие этим изменениям.

Чтобы добиться согласованного управления пользователями и ресурсами, и обеспечить слаженное и эффективное изменение методов доступа к необходимым для бизнеса приложениям и данным, требуется унифицированное средство, которое способно:

- поддерживать множество платформ (ОС, файловых систем, баз данных);
- поддерживать как приложения, так и системы;
- иметь возможность физического масштабирования на тысячи систем;
- обеспечивать быстрое администрирование тысяч пользователей;
- обеспечивать возможность администрирования на основе политики организации (policy-based), определяющей права доступа в зависимости от положения или роли пользователя в корпорации.

Таким требованиям удовлетворяют интегрированные системы управления. По данным IDC, выполнение административных операций в

интегрированной среде позволяет более чем вдвое снизить затрачиваемое на эти задачи время для 100 пользователей. Для примера: управляющее приложение TME 10 User Administration задает единый шаблон для учетных данных пользователя на разнородных системах и позволяет модифицировать эти данные с помощью только одной операции.

### 3. Канал **Trace Mode 5** (значения канала, обработка данных в канале, классификация каналов).

Рассмотрим теперь элементарное звено информационной структуры проекта ТРЕЙС МОУД. Оно называется **каналом**.

**Канал** - это базовое понятие системы. Данные с внешних устройств записываются в каналы. Данные из каналов посылаются на внешние устройства и отображаются на экране монитора. Значения из каналов записываются в архивы и отчеты. В каналах осуществляется преобразование данных. С помощью системных каналов можно управлять выводимой на экран информацией, звуковыми эффектами, архивами и т.д., то есть всей системой.

Совокупность всех каналов - **база каналов** - составляет математическую основу программного обеспечения каждого узла проекта.

#### Определение канала

Канал – это структура, состоящая из набора **переменных** и **процедур**, имеющая **настройки** на внешние данные, **идентификаторы** и **период** пересчета ее переменных.

Идентификаторами канала являются: **имя**, **комментарий** и **кодировка**. Кроме того, каждый канал имеет числовой идентификатор, используемый внутри системы для ссылок на этот канал.

Среди переменных канала выделяются четыре основных значения: **входное**, **аппаратное**, **реальное** и **выходное**. С помощью **настроек** входное значение канала связывается с источником данных, а выходное – с приемником. С помощью **процедур** входное значение канала преобразуется в аппаратное, реальное и выходное.

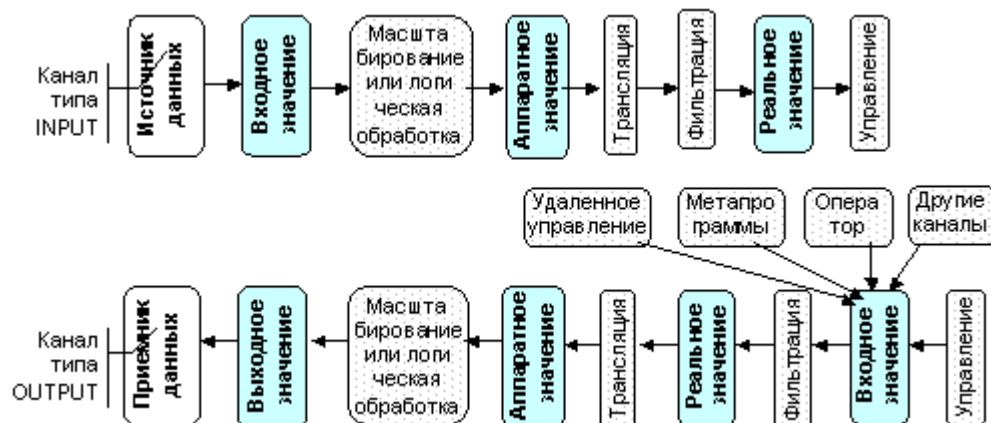
Процедурами канала являются: **масштабирование** (умножение и смещение), **фильтрация** (подавление пиков, апертюра и сглаживание), **логическая обработка** (предустановка, инверсия, контроль сочетаемости), **трансляция** (вызов внешней программы) и **управление** (вызов внешней программы).

Кроме основных значений канал имеет дополнительные переменные: шесть границ, гистерезис, настройки процедур обработки, начальные параметры, флаги архивирования и др.

Переменные, настройки и идентификаторы канала образуют список его **атрибутов**. Часть из них задается в редакторе базы каналов и не может быть изменена в реальном времени. Другие могут иметь начальные значения и доступны для изменения.

В зависимости от направления движения информации, т.е. от внешних источников (данные с контроллеров, УСО или системные переменные) в канал или наоборот, каналы

подразделяются на входные (тип INPUT) и выходные (тип OUTPUT). Ниже на рисунке показаны структуры канала обоих типов.



## Значения канала

### [Точка ввода-вывода](#)

### [Каналы типа INPUT](#)

### [Каналы типа OUTPUT](#)

Каждый канал имеет четыре основных значения: **входное**, **аппаратное**, **реальное** и **выходное**. Они обозначаются следующим образом:

**In** – входное;

**A** – аппаратные;

**R** – реальные;

**Q** – выходное.

Значения канала могут иметь один из следующих форматов:

число с плавающей точкой одинарной точности;

16-битовое целое число.

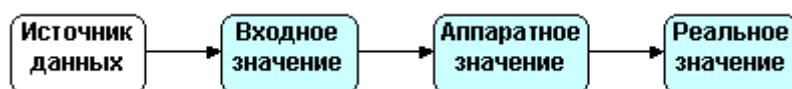
Первый формат используется для аналоговых переменных, второй – для дискретных.

### Точка ввода-вывода

Под точкой ввода/вывода в ТРЕЙС МОУД понимается входное или выходное значение канала, связанное с внешним источником/приемником данных по стандартным протоколам ТРЕЙС МОУД, Windows или по протоколам драйверов УСО (устройств сопряжения с объектом). При этом такое значение канала типа FLOAT соответствует 1 точке ввода/вывода, а типа HEX – **n** точкам, где **n** – разрядность значения.

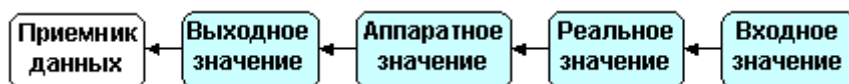
### Каналы типа INPUT

Входной канал запрашивает данные у внешнего источника (контроллер, другой МРВ и пр.) или значение системных переменных (счетчик ошибок, длина архива и пр.). Полученное значение поступает на вход канала и далее пересчитывается в аппаратное и реальное значения. Выходные значения в них не используются.



### Каналы типа OUTPUT

Выходной канал передает данные приемнику. Приемник может быть внешним (значение переменной в контроллере, в другом МРВ и пр.) или внутренним - одна из системных переменных (номер проигрываемого звукового файла, номер экрана, выводимого на монитор, и пр.). И внешние и внутренние приемники данных связываются с выходными значениями каналов.



### Входные значения

Входное значение каналов типа INPUT может формироваться одним из следующих способов:

данными от внешних источников (управляющие контроллеры, УСО, данные с удаленных узлов и пр.);

данными, запрашиваемыми у системы (системные переменные, значения других каналов и пр.).

У каналов типа OUTPUT их входное значение формируется одним из следующих способов:

процедурой **управление** данного канала;

процедурами **управление** или **трансляция** других каналов;

Метапрограммой на языке Техно IL (см. ниже);

Каналом удаленного узла (например, по сети);

Оператором с помощью управляющих графических форм.

### Аппаратное значение

Это значение у каналов типа INPUT формируется **масштабированием** (логической обработкой для дискретных каналов) входных значений. У каналов типа OUTPUT аппаратное значение получается из реального процедурой **трансляции**.

Аппаратные значения каналов имеют такое название, поскольку в них удобно получать величины унифицированных сигналов, с которыми работает аппаратура ввода/вывода (4-20 мА, 0-10 В и пр.).

### **Реальное значение**

Эти значения предназначены для хранения значений контролируемых параметров или сигналов управления в реальных единицах (например, кг/час, °С, % и пр.).

Для входных каналов (тип INPUT) реальные значения формируются из аппаратных процедурами **трансляции** и **фильтрации**. Если канал является выходным (тип OUTPUT), то его реальное значение получается из входного после **фильтрации**.

### **Выходные значения**

Это значение определено только для каналов типа OUTPUT. Оно пересчитывается из аппаратного значения. В аналоговых каналах используется процедура **масштабирования**. Для каналов обрабатывающих дискретные сигналы выходное значение формируется из аппаратного **логической обработкой**.

### **Обработка данных в канале**

Для обработки данных и формирования своих значений каналы ТРЕЙС МОУД имеют следующие процедуры:

масштабирование;

логическая обработка;

трансляция;

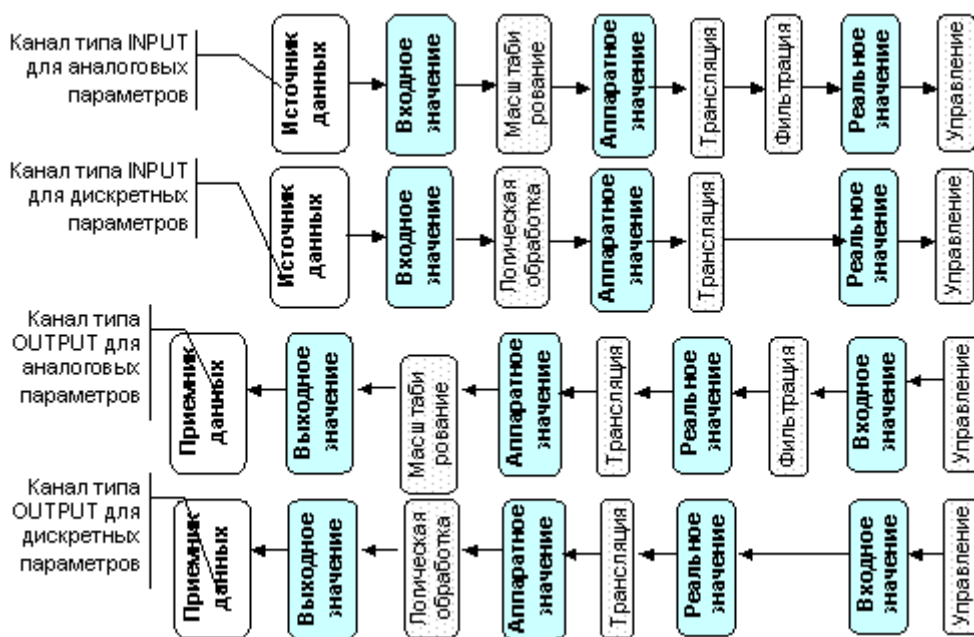
фильтрация;

управление.

Набор процедур в канале зависит от формата данных. Каналы, работающие с аналоговыми переменными, используют процедуры **масштабирование**, **трансляция**, **фильтрация** и **управление**. В каналах обрабатывающих дискретные параметры используются **логическая обработка**, **трансляция** и **управление**.

**Внимание!** В одном канале Микро МРВ процедуры ТРАНСЛЯЦИЯ и УПРАВЛЕНИЕ использовать одновременно нельзя.

Порядок следования и содержание процедур может меняться в зависимости от типа канала (входной или выходной). На рисунке представлены четыре возможные схемы каналов.



В каналах типа INPUT их процедуры обеспечивают первичную обработку данных (исправление ошибок датчиков, масштабирование, коррекция температуры холодных спаев термопар и т. д.). В каналах типа OUTPUT процедуры преобразуют величину управляющего воздействия из реального формата (проценты, амперы и пр.) к виду, воспринимаемому внешними устройствами. Подробно настройка первичной и выходной обработки будет рассмотрена в разделе, посвященном редактору базы каналов.

Рассмотрим более подробно назначение процедур канала, и какие преобразования они выполняют.

### Масштабирование

Эта процедура используется только в каналах, работающих с аналоговыми переменными. Она включает в себя две операции: **умножение** и **смещение**. Последовательность этих операций меняется в зависимости от типа канала:

у каналов типа INPUT входное значение умножается на заданный множитель и к полученному результату добавляется величина смещения. Результат присваивается аппаратному значению канала.

У каналов типа OUTPUT к аппаратному значению добавляется величина смещения, затем эта сумма умножается на заданный множитель, а результат присваивается выходному значению канала.

### Логическая обработка

Эта процедура определена только для каналов, обрабатывающих дискретные сигналы. В каналах типа INPUT она по входному значению формирует аппаратное, а если тип OUTPUT, то по аппаратному выходное. Канал работает с упакованными дискретными сигналами (до 16 сигналов). Для обработки этих сигналы можно задать три маски. С их помощью выполняются следующие операции:

**предустановка** – логическое сложение с маской. Эта операция используется когда, независимо от значения дискретных сигналов, их величину в логическом управлении надо принять равной 1. В маске указываются биты, которые надо установить равными 1;

**инверсия** – исключающее логическое сложение с маской. В ней указываются инвертируемые биты;

**анализ на сочетаемость** – логическое умножение на маску и сравнение результата со значением маски: их равенство считается ошибочной ситуацией. Маска указывает биты, которые не могут одновременно быть равны 1 (например, сигналы с концевых выключателей открытия и закрытия задвижки).

Имеется возможность распаковать дискретные сигналы по отдельным каналам. Это будет описано ниже.

## Трансляция

Эта процедура определена для всех каналов независимо от их типа и вида представления. У входных каналов процедура трансляции преобразует **аппаратное** значение в **реальное**, а для выходных – наоборот. Для этого вызывается FBD-программа. Вызываемая программа выбирается при настройке процедуры. Так же при настройке процедуры входные и выходные аргументы выбранной программы связываются с атрибутами текущего канала, а так же любых других каналов из текущей базы. Поэтому процедура трансляции одного канала может так же использоваться для формирования значений других каналов.

## Пример

Использование процедуры трансляции.

Необходимо измерять расход вещества, транспортируемого по трубопроводу, и интегрировать его по времени для расчета технико-экономических показателей производства. На трубопроводе установлен датчик скорости потока.

Для решения этой задачи потребуется один канал типа **INPUT**. Его аппаратное значение свяжем с данными, поступающими от датчика скорости потока (адресация каналов описана в следующем разделе).

Настроим коэффициенты масштабирования и дрейфа нуля исходя из геометрических характеристик трубопровода и физических свойств потока для перевода измеренной скорости в величину расхода. Затем создадим программу, интегрирующую входную величину и записывающую результат в выходную переменную. Далее эту программу надо установить для процедуры трансляция данного канала (написание программ для процедур канала будет рассмотрено ниже).

При такой конфигурации во **входном** значении канала будет находиться информация о скорости потока, в **аппаратном** - величина расхода вещества, а в **реальном** - количество прошедшего по трубе вещества.

## Фильтрация



Данная процедура присутствует только у аналоговых каналов. Набор выполняемых ею операций отличается для входных и выходных каналов.

У каналов типа INPUT фильтрация выполняется после процедуры трансляции до формирования реального значения. Фильтрация включает в себя следующие операции:

подавление случайных всплесков в тракте измерения;

подавление малых колебаний значения канала;

экспоненциальное сглаживание;

контроль шкалы – отслеживание выхода реального значения канала за установленные границы шкалы (см. ниже);

У каналов типа OUTPUT данная процедура формирует реальное значение по входному значению. При этом выполняются следующие операции:

ограничение скорости изменения реального значения;

подавление малых колебаний значения канала;

экспоненциальное сглаживание;

контроль шкалы – обрезание величины управляющего воздействия до границ шкалы канала (см. ниже).

## **Управление**

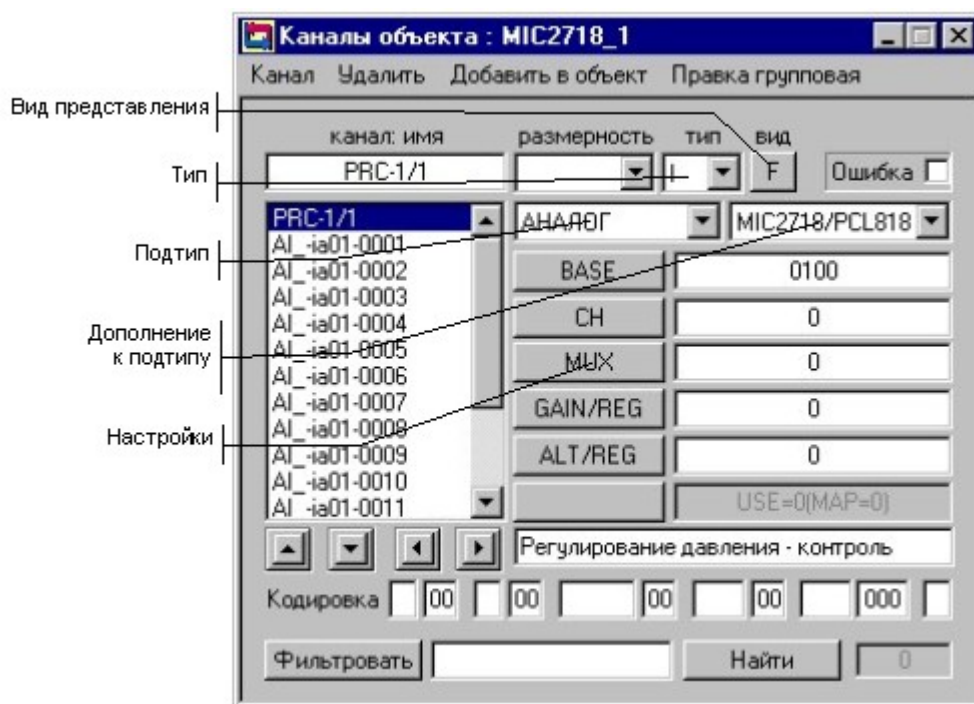
Эта процедура определена для всех каналов. Она реализует функцию управления. С ее помощью можно вызвать **FVD**-программу, в которой можно запрограммировать требуемые алгоритмы управления. В качестве аргументов программе могут передаваться значения и атрибуты любых каналов из текущей базы. Эти аргументы могут быть как входными, так и формируемыми.

Формально процедура управление связана с каналом только циклом пересчета. Она может вообще никак не участвовать в формировании его значений, а управлять другими каналами. Такая ситуация часто наблюдается при использовании процедуры **управление** на каналах типа INPUT.

## **Классификация каналов**

Каналы могут принимать данные с внешних устройств или управлять ими, контролировать или управлять работой системы и пр. Назначение канала определяется его **типом, подтипом и дополнениями к подтипу**. Адресация к данным в рамках его назначения осуществляется с помощью **настроек**. Каналы могут обрабатывать аналоговые или дискретные сигналы. Это устанавливается с помощью параметра **вид представления**.

Эти параметры канала настраиваются в диалоге **Каналы объекта**. Вход в него осуществляется из окна редактирования базы каналов по двойному ЛК на изображении объекта. Области настройки перечисленных параметров показаны на следующем рисунке.



### Тип канала

Тип определяет направление движения информации в канале. Существует два типа: входной (**INPUT**) и выходной (**OUTPUT**).

Входные значения каналов типа INPUT связываются с источниками данных. Вид источника определяется подтипом и дополнением к подтипу канала. Это могут быть данные, получаемые от контроллеров, по сети, от плат ввода измерительных сигналов и пр.

Каналы типа OUTPUT передают свои выходные значения приемникам данных. Эти значения могут управлять системными задачами, передаваться по сети на удаленные узлы или внешним устройствам. Приемник данных определяется подтипом и дополнением к подтипу канала.

В поле задания типа канала диалог **Каналы объекта** для идентификации типа используется один символ:

**I** - тип INPUT;

**O** - тип OUTPUT.

В большинстве случаев тип канала выставляется автоматически при задании подтипа и дополнения к подтипу. Однако иногда его надо устанавливать вручную.

### Подтип канала

Подтип канала указывает класс источников или приемников данных, с которыми будет связываться канал.

Для каналов типа INPUT подтип характеризует получаемую ими информацию (АНАЛОГ - значение АЦП, считанное с платы УСО, СИСТЕМНЫЙ - состояние системы, СВЯЗЬ -

данные с удаленных узлов проекта и пр.). Каналы OUTPUT имеют тот же набор подтипов, что и каналы INPUT. Однако для них подтип определяет класс приемников, а не источников данных (АНАЛОГ - значение ЦАП, СИСТЕМНЫЙ - состояние системы, СВЯЗЬ - значения управляемых каналов на удаленных узлах проекта и пр.).

Всего существует шестнадцать подтипов каналов. Все они могут задаваться как для входных, так и для выходных каналов.

### **Дополнение к подтипу**

Все подтипы каналов имеют дополнение к подтипу. С помощью дополнений уточняется тип источника или приемника данных. Например, для канала подтипа СВЯЗЬ с помощью дополнения задается среда передачи: локальная сеть, последовательный интерфейс или коммутируемые линии.

### **Настройки**

Для адресации данных в рамках установленного типа источника или приемника данных используются настройки канала. Их число зависит от подтипа и дополнения к подтипу. Примером таких настроек могут быть базовый адрес платы УСО и номер канала ввода на этой плате.

### **Вид представления**

Канал может работать с аналоговыми параметрами (значения температур, расходов, напряжений и т. д.) и с дискретными (сигналы от концевых выключателей, магнитных пускателей, пороговых датчиков и т. д.). Тип данных, с которыми работает канал, определяется его **видом представления**:

число с плавающей точкой одинарной точности;

16-битовое целое число.

Первый из них предназначен для аналоговых переменных, второй – для дискретных. Таким образом, канал может обрабатывать либо один аналоговый сигнал, либо до 16 дискретных.

Вид представления устанавливается нажатием ЛК в области, показанной на рисунке в начале раздела. По умолчанию каналы имеют вид представления - число с плавающей точкой, что индицируется буквой **F** (float). При нажатии на ней ЛК буква **F** сменится на букву **H** (hex), что соответствует виду представления для работы с дискретными сигналами.

Списки атрибутов у каналов с разным видом представления отличаются.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 3. [Редактор базы каналов Trace Mode 5](#).  
Справочная система Trace Mode 5, раздел: «Редактор базы каналов», глава:  
«Автопостроение баз каналов».**

## ЛЕКЦИЯ 4 СОВРЕМЕННЫЕ SCADA-СИСТЕМЫ

### Введение

Автоматизированная система управления - представляет собой совокупность управляемого объекта и автоматизированных управляющих устройств, состоит из системы автоматики и системы операторского управления.

### SCADA - Supervising Control and Data Acquisition

SCADA система (система диспетчерского управления и сбора данных) - это совокупность устройств управления и мониторинга, а также средств их взаимодействия с технологическим объектом.

Любая SCADA - система включает в себя инструментальные средства для создания пользовательского интерфейса рабочих мест операторов, исполнительные модули и средства взаимодействия с технологическим процессом.

Популярные на западном и российском рынках SCADA-системы, имеющие поддержку в России [1]:

- Factory Link (United States DATA Co., USA );
- InTouch (Wonderware, USA);
- Genesis (Iconics, USA);
- WinCC (Siemens, Germany);
- Trace Mode (Ad Astra, Россия);
- Master-Scada (InSAT)
- Контур (Россия)
- RSView (Rockwell Software Inc, USA);
- LabVIEW, BridgeVIEW, LabVIEW RT, Lookout (National Instruments, USA) и др.
- 

SCADA-системы, прежде всего, предназначены для получения и визуализации информации от программируемых логических контроллеров (ПЛК), плат ввода-вывода информации, распределенных систем управления.

### **1. Основные возможности и характеристики современных SCADA-систем**

### **SCADA – системы выполняют следующие функции:**

1. Сбор, первичную обработку и накопление информации о параметрах технологического процесса и состоянии оборудования от промышленных контроллеров и других цифровых устройств, непосредственно связанных с технологической аппаратурой;
2. Отображение информации о текущих параметрах технологического процесса на экране ПЭВМ в виде графических мнемосхем;
3. Отображение графиков текущих значений технологических параметров в реальном времени за заданный интервал;
4. Обнаружение критических (аварийных) ситуаций;
5. Вывод на экран ПЭВМ технологических и аварийных сообщений, их архивирование;
6. Сохранение истории изменения значений аналоговых и дискретных переменных;
7. Дистанционное управление технологическим процессом;
8. Предоставление данных о параметрах технологического процесса для их использования в системах управления предприятием.

### **Функциональные возможности**

1. Разработка архитектуры всей системы автоматизации (на этом этапе определяется функциональное назначение каждого узла системы автоматизации).
2. Решение вопросов, связанных с возможной поддержкой распределенной архитектуры, необходимостью введения узлов с горячим резервированием и т.п.
3. Создание прикладной системы управления для каждого узла, где специалист в области автоматизируемых процессов наполняет узлы архитектуры алгоритмами, совокупность которых позволяет решать задачи автоматизации.

4. Приведение параметров прикладной системы в соответствие с информацией, которой обмениваются устройства нижнего уровня (ПЛС, АЦП, ЦАП) с внешним миром (датчиками температуры, давления и др.).

5. Отладка созданной прикладной программы в режиме эмуляции и реальном режиме.

## **Технические характеристики**

### ***1. Программно-аппаратные платформы.***

Анализ перечня таких платформ необходим, поскольку от него зависит распространение SCADA-системы на имеющиеся вычислительные средства, а также оценивание стоимости ее эксплуатации. Подавляющее большинство SCADA-систем реализовано на MS Windows-платформах (Windows NT).

### ***2. Имеющиеся средства сетевой поддержки.***

Для эффективного функционирования системы автоматизации распределенных объектов SCADA-система должна обеспечивать высокий уровень сетевого сервиса. Необходима поддержка сетевых сред с использованием стандартных протоколов (Netbios, TCP/IP и др.), а также наиболее популярных сетевых стандартов из класса промышленных интерфейсов (Profibus, Canbus, LON, Modbus и т.д.).

### ***3. Встроенные командные языки.***

Большинство SCADA-систем имеют встроенные языки высокого уровня, Basic-подобные языки, для создания фрагментов алгоритма, необходимых в решении задачи управления.

### ***4. Поддерживаемые БД.***

Практически во всех SCADA-системах осуществлена поддержка SQL-синтаксиса, не зависящего от типа БД, что позволяет создавать независимые программы для анализа информации и использовать уже имеющееся ПО, ориентированное на обработку данных.

### ***5. Графические возможности.***

Функционально графические интерфейсы SCADA-систем весьма похожи. В каждой из них существует графический объектно-ориентированный редактор с определенным набором анимационных функций. Используемая векторная графика дает возможность

осуществлять широкий набор операций над выбранным объектом, а также быстро обновлять изображение на экране средствами анимации. Крайне важен вопрос о поддержке в рассматриваемых системах стандартных функций GUI (Graphic Users Interface). Поскольку большинство рассматриваемых SCADA-систем работает под управлением Windows, это и определяет тип используемого GUI.

## **Эксплуатационные характеристики**

### ***1. Удобство использования.***

Сервис, предоставляемый SCADA-системами на этапе разработки ППО, обычно очень развит. Почти все они имеют Windows-подобный пользовательский интерфейс, что во многом повышает удобство их использования, как в процессе разработки, так и в период эксплуатации прикладной задачи.

### ***2. Наличие и качество поддержки.***

Возможны следующие уровни поддержки: услуги фирмы-разработчика, обслуживание региональными представителями фирмы-разработчика, взаимодействие с системными интеграторами, русификация программ и документации, горячая линия и решение проблем, связанных с индивидуальными требованиями заказчика и др.

## **2. Компоненты систем контроля и управления и их назначение**

Многие проекты автоматизированных систем контроля и управления (СКУ) для большого спектра областей применения позволяют выделить обобщенную схему их реализации, представленную на рис.1.

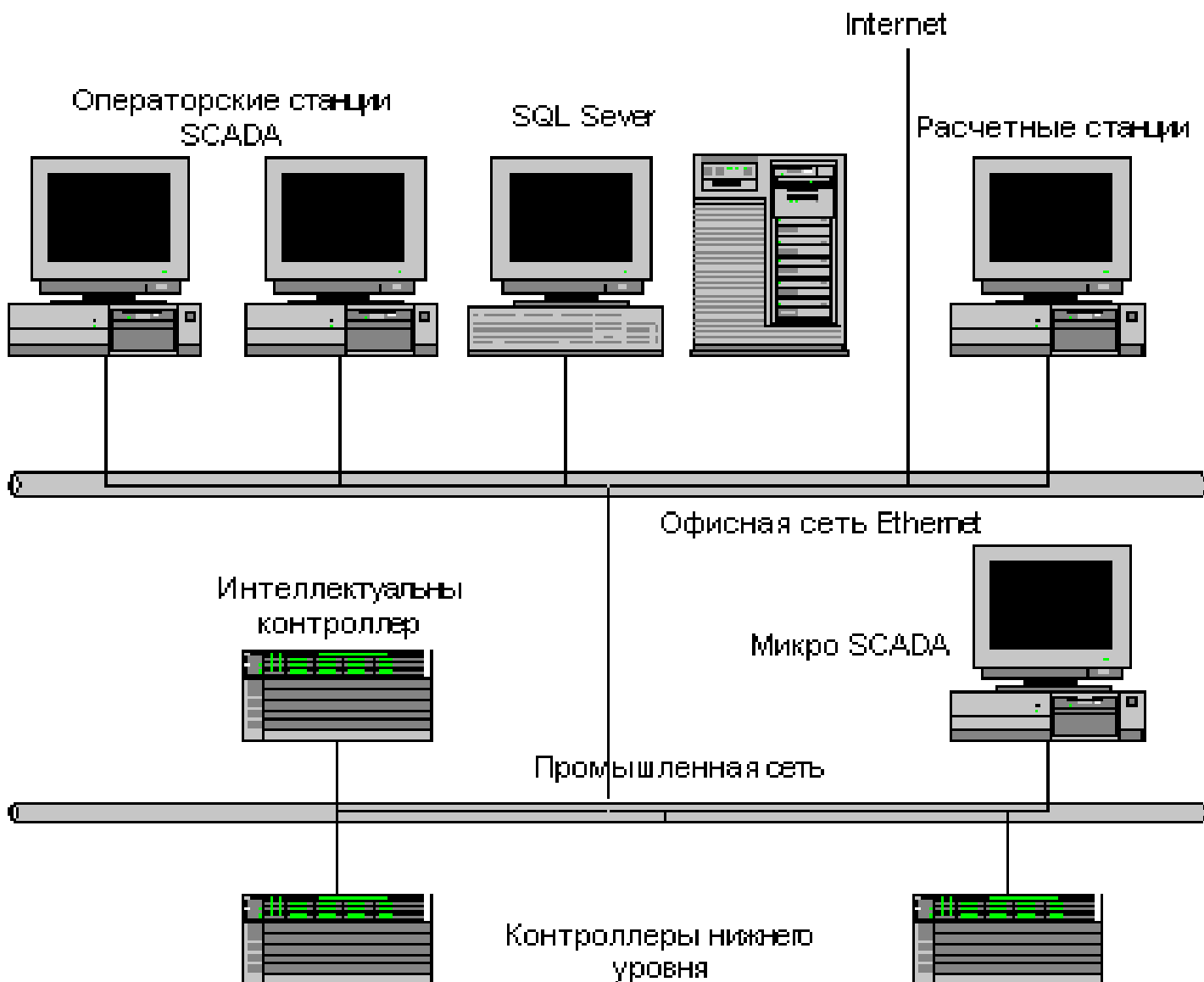


Рис.4. Обобщенная схема системы контроля и управления.

Как правило, это двухуровневые системы, так как именно на этих уровнях реализуется непосредственное управление технологическими процессами. Специфика каждой конкретной системы управления определяется используемой на каждом уровне программно - аппаратной платформой.

### **Нижний уровень - уровень объекта (контроллерный)**

Включает различные датчики для сбора информации о ходе технологического процесса, электроприводы и исполнительные механизмы для реализации регулирующих и управляющих воздействий.

Датчики поставляют информацию локальным программируемым логическим контроллерам (PLC - Programming Logical Controller), которые могут выполнять следующие функции:



- сбор и обработка информации о параметрах технологического процесса;
- управление электроприводами и другими исполнительными механизмами;
- решение задач автоматического логического управления и др.

Так как информация в контроллерах предварительно обрабатывается и частично используется на месте, существенно снижаются требования к пропускной способности каналов связи.

К аппаратно-программным средствам контроллерного уровня управления предъявляются жесткие требования по надежности, времени реакции на исполнительные устройства, датчики и т.д. Программируемые логические контроллеры должны гарантированно откликаться на внешние события, поступающие от объекта, за время, определенное для каждого события.

Для критичных с этой точки зрения объектов рекомендуется использовать контроллеры с операционными системами реального времени (ОСРВ). Контроллеры под управлением ОСРВ функционируют в режиме жесткого реального времени.

Разработка, отладка и исполнение программ управления локальными контроллерами осуществляется с помощью специализированного программного обеспечения, широко представленного на рынке.

### **Верхний уровень - уровень оперативного управления (АРМ)**

Информация с локальных контроллеров может направляться в сеть диспетчерского пункта непосредственно, а также через контроллеры верхнего уровня (см. рис.1). В зависимости от поставленной задачи контроллеры верхнего уровня (концентраторы, интеллектуальные или коммуникационные контроллеры) реализуют различные функции. Некоторые из них перечислены ниже:

- сбор данных с локальных контроллеров;
- обработка данных, включая масштабирование;
- поддержание единого времени в системе;
- синхронизация работы подсистем;

- организация архивов по выбранным параметрам;
- обмен информацией между локальными контроллерами и верхним уровнем;
- работа в автономном режиме при нарушениях связи с верхним уровнем;
- резервирование каналов передачи данных и др.

Верхний уровень - диспетчерский пункт (ДП) - включает, прежде всего, одну или несколько станций управления, представляющих собой автоматизированное рабочее место (АРМ) диспетчера/оператора.

Здесь же может быть размещен сервер базы данных, рабочие места (компьютеры) для специалистов и т. д. Часто в качестве рабочих станций используются ПЭВМ типа IBM PC различных конфигураций.

Станции управления предназначены для отображения хода технологического процесса и оперативного управления. Эти задачи и призваны решать SCADA - системы.

Рассматривая обобщенную структуру систем управления, следует ввести и еще одно понятие - Micro-SCADA. Micro-SCADA - это системы, реализующие стандартные (базовые) функции, присущие SCADA - системам верхнего уровня, но ориентированные на решение задач автоматизации в определенной отрасли (узкоспециализированные). В противоположность им SCADA - системы верхнего уровня являются универсальными.

- Все компоненты системы управления объединены между собой каналами связи. Обеспечение взаимодействия SCADA - систем с локальными контроллерами, контроллерами верхнего уровня, офисными и промышленными сетями возложено на так называемое коммуникационное ПО. Это достаточно широкий класс программного обеспечения, выбор которого для конкретной системы управления определяется многими факторами, в том числе и типом применяемых контроллеров, и используемой SCADA - системой.
- Большой объем информации, непрерывно поступающий с устройств ввода/вывода систем управления, предопределяет наличие в таких системах баз данных (БД). Основная задача баз данных - своевременно обеспечить пользователя всех уровней управления требуемой информацией. Но если на верхних уровнях АСУ эта задача решена с помощью традиционных БД, то этого не скажешь об уровне АСУ ТП. До недавнего времени регистрация информации в реальном времени решалась на базе ПО интеллектуальных контроллеров и SCADA -

систем. В последнее время появились новые возможности по обеспечению высокоскоростного хранения информации в БД.

### 3. Атрибуты канала

- имя и индекс канала;
- период и фаза работы канала;
- время изменения;
- атрибуты СПАД, регистратор, отчёт;
- состояние канала;
- начальное значение канала;
- флаги архивирования.

Выше были описаны основные составляющие канала ТРЕЙС МОУД - значения и процедуры. Теперь давайте рассмотрим, как специфицируется канал, задаются режимы его работы, какие атрибуты определены для него и каково их назначение.

Любой канал обязательно имеет следующий набор атрибутов:

имя;

тип канала;

подтип канала;

дополнение к подтипу канала ;

вид представления;

период и фаза;

подключение;

состояние канала;

начальное значение;

кодировка;

размерность;

СПАД, регистратор, отчет тревог, атрибуты;

время изменения.

Кроме того, для каждого канала можно установить ряд флагов, определяющих режимы пересчета и архивирования данных:

В сеть;

Доступ;

Отработать;

СПАД;

Тревоги;

Регистратор;

Атрибуты.

Атрибут **вид представления** указывает тип данных, с которыми работает канал: аналоговые или дискретные. Набор атрибутов канала зависит от его вида представления. Канал, обрабатывающий аналоговые переменные имеет следующие атрибуты:

четыре аварийные границы;

гистерезис;

шкала - две границы;

коэффициент сглаживания;

множитель;

дрейф нуля;

величина апертуры;

порог фильтрации пиков;

Кроме того, эти каналы имеют флаги включения операций: **сглаживание, апертура, фильтрация пиков, умножение, смещение** (дрейф нуля), **контроль границ шкалы**.

Каналы, обрабатывающие дискретные переменные, имеют следующие специфичные атрибуты:

маска контроля сочетаемости;

маска инверсии;

маска предустановки;

Эти каналы имеют флаги включения следующих операций: **контроль сочетаемости, инверсия, предустановка**.

Кроме перечисленных выше, канал обладает еще рядом атрибутов. Они вычисляются системой или формируются по заданным алгоритмам. К таким атрибутам относятся:

интервал;

тенденция;

достоверность;

события.

Рассмотрим теперь более подробно назначение каждого из перечисленных выше атрибутов канала ТРЕЙС МОУД.

### **Имя и индекс канала**

Основным идентификатором канала является его имя. Оно используется для ссылок на его значения. По имени канала так же привязываются динамические формы в редакторе представления данных.

При создании канала его имя формируется автоматически. При автопостроении оно отражает источник данных. Например, имя канала, связанного с пятым каналом платы аналоговые входы, расположенной в первом посадочном месте контроллера, будет **AI - pp01-0005**. Если канал создан автопостроением запроса данных с удаленных узлов, то его имя воспроизводит имя канала-источника.

При создании канала вручную ему присваивается следующее имя:

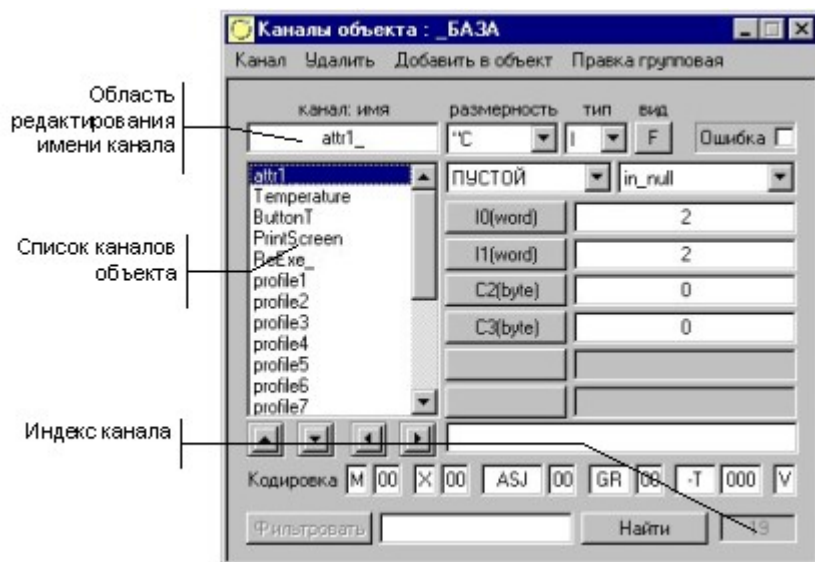
**new-xx00-`<число>`**

где

**`<число>`** – индекс канала.

Индекс – это номер канала по порядку в объекте **БАЗА**. Индекс автоматически присваивается всем каналам (созданным как вручную, так и при автопостроении). Этот параметр является числовым идентификатором канала и не изменяется при любых операциях с данным каналом (переименование, переопределение типа, подтипа и т.п.). При удалении канала его индекс остается свободным. Вновь создаваемому каналу присваивается младший из свободных индексов. Индекс канала отображается в правом нижнем окне диалога **Каналы объекта** редактора базы каналов. Каналы пересчитываются последовательно, начиная с канала, имеющего младший индекс.

Чтобы изменить имя канала, надо открыть диалог **Каналы объекта** того объекта, где он находится. Далее следует выделить этот канал в списке внести требуемые изменения.



В имени канала можно использовать любые символы. Однако, не рекомендуется использовать символы математических операций, знаки препинания и пробелы. Это важно если на канал организуются внешние ссылки из других приложений и метапрограмм.

Имя канала не может быть менее 6 символов. В противном случае оно дополняется до этого размера символами подчеркивания. Максимальный размер имени составляет 13 символов.

## Период и фаза работы канала

### [Особенности отслеживания периодов каналов](#)

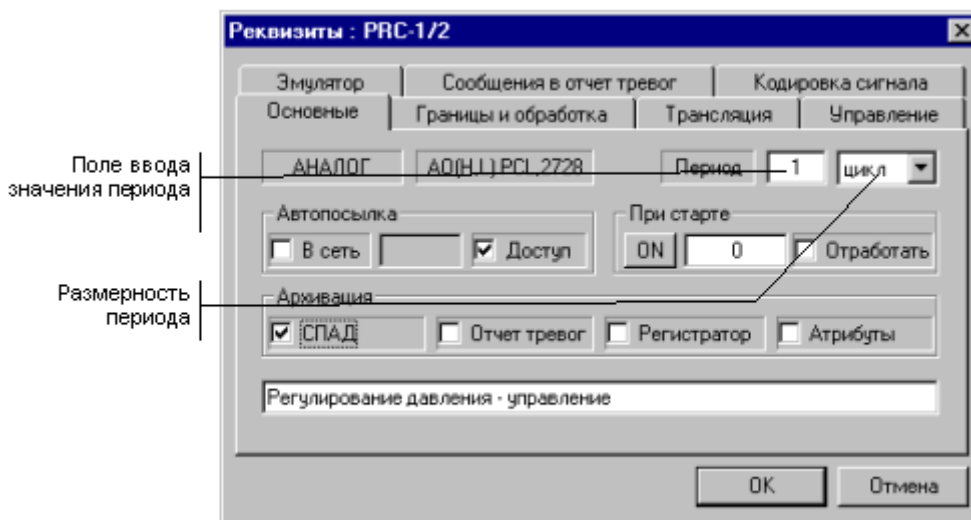
### [Подходы к заданию периодов каналов](#)

### [Оперативная коррекция периода канала](#)

Период канала определяет частоту обновления данных и пересчета его значений. Значение периода может устанавливаться либо в единицах времени (секунды, минуты, часы), либо в периодах пересчета базы каналов (циклах системы).

Время, отводимое на цикл системы, называется периодом пересчета. Этот параметр определяет частоту выполнения основных системных задач (обмен данными с контроллерами, пересчет базы каналов, перерисовка экрана, сетевой обмен и пр.). Если задать период пересчета недостаточным для выполнения всех задач, то наименее приоритетные из них (например, обновление экрана монитора) будут тормозиться. Однако большое значение периода пересчета уменьшает скорость реакции системы. Таким образом, период пересчета надо задавать исходя из выполнения всех ее задач, сохраняя при этом требуемую реактивность системы.

Значение и размерность периода вводятся в разделе **Период** бланка **Основные** диалога **Реквизиты**, показанного на рисунке.



Вход в этот диалог производится по двойному нажатию ЛК на имени требуемого канала в списке диалога **Каналы объекта**.

Размерность периода канала выбирается из следующего списка:

- цикл** – период в циклах;
- сек** – период в секундах;
- мин** – период в минутах;
- час** – период в часах;
- флаг** – период в циклах с отключением пересчета по системному флагу;
- F1** – период в циклах с отставанием на 1 цикл;
- F2** – период в циклах с отставанием на 2 цикла;
- F3** – период в циклах с отставанием на 3 цикла;
- F4** – период в циклах с отставанием на 4 цикла;
- на старте** – канал обрабатывается один раз при запуске;
- быстрый** – канал работает вне цикла системы.

Значение периода может изменяться в диапазоне от 1 до 100. По умолчанию оно задается в циклах и равно 1.

### Особенности отслеживания периодов каналов

Для каналов, период которых задан в циклах (кроме **F1, ..., F4**), необходимость пересчета определяется по равенству 0 остатка от деления величины **индекс пересчета** на период. Первый увеличивается на 1 на каждом цикле системы и сбрасывается в 1 при достижении значения 100. Его текущую величину можно контролировать с помощью канала подтипа **СИСТЕМНЫЙ** с дополнением **индекс пересчета**.

Для каналов, имеющих тип периода **F1**, ..., **F4**, условие пересчета записывается следующим образом:

**(Индекс пересчета % 5 = I) & (Индекс пересчета % FRQ = 0)**

где

**i** – величина фазы (1, 2, 3 или 4);

**FRQ** – величина периода.

Исходя из этого условия, значение периода канала 1 и размерность **F1** устанавливают его пересчет один раз в 5 циклов с отставанием на 1 цикл от канала с периодом равным 5 и размерностью **цикл**. Каналы с периодом **F2** в этих же условиях будут отставать на 2 цикла, **F3** – на 3 цикла, а **F4** – на 4 цикла. Значение периода 2 задает для этих каналов пересчет раз в 10 циклов и т. д.

Введение фазы позволяет распределить нагрузку по пересчету каналов с одинаковым значением периода на разные циклы системы. Рассмотрим пример использования распределения пересчета.

### Пример

В базе присутствуют 10000 каналов. Если их период установлен равным 5, то они пересчитываются один раз в пять циклов на одном цикле. В течение четырех циклов нагрузка по пересчету нулевая, а на одном - максимальная. При этом можно перегрузить систему вычислениями на пятом цикле.

Если разбить каналы на пять групп, по 2000 в каждой. У первой оставить период равным 5, а у остальных – равным 1 и использовать размерности **F1**, ..., **F4**, то нагрузка по пересчету равномерно распределяется во времени. В этом случае на каждом цикле системы будут пересчитываться только 2000 каналов. В такой ситуации, как и в первом случае, данные во всех каналах будут обновляться с одинаковой частотой. Однако обновление будет сдвинуто по фазе для каждой группы.

При периоде, заданном в **секундах**, **минутах** или **часах** (не более 23 ч), значения канала пересчитываются на первом цикле системы по истечении периода. Такой пересчет привязан к астрономическому времени.

В Windows NT: если период равен 24 часам, то канал будет пересчитываться 1 раз в сутки в 00:00:00. Чтобы канал пересчитывался 1 раз в сутки в 01:00:00 (02:00:00,...12:00:00), нужно задать период обработки равным 25 ч (26,...36 ч). При этом в базе каналов запоминается, что этот канал обрабатывается 1 раз в сутки, а для изменения момента обработки в процессе работы надо задавать период в естественных единицах – соответственно от 1 до 12 ч.

Привязка к астрономическому времени осуществляется при первом равенстве 0 соответствующего временного диапазона. Например, если период задан в секундах, то его привязка к астрономическому времени осуществляется при переходе к следующей минуте.



Если величина периода не кратна соответствующему временному диапазону и меньше его половины, например 25 секунд, то канал будет пересчитываться всегда на 0-й, 25-й и 50-й секунде астрономического времени.

Если величина периода больше половины соответствующего временного диапазона, например 40 секунд, то канал будет пересчитываться всегда на 40-й секунде астрономического времени (1 раз в минуту).

Каналы, имеющие тип периода **флаг**, пересчитываются так же, как и каналы с периодом **цикл**. Однако их пересчет блокируется флагом пересчета, который устанавливается каналом типа **OUTPUT** подтипа **СИСТЕМНЫЙ** с дополнением **флаг пересчета**.

### **Подходы к заданию периодов каналов**

Для каналов, получающих данные от аппаратуры ввода/вывода, их период задает частоту опроса технологических параметров. Для выходных каналов период канала задает частоту формирования управляющих сигналов.

В принципе можно задать одинаковую частоту опроса для всех контролируемых технологических параметров, ориентируясь на наиболее динамичные, или просто принять значения периодов всех каналов по умолчанию, что соответствует максимальной частоте. Однако в этом случае при создании крупных систем с большим количеством контролируемых параметров может не хватить ресурсов компьютера для задач обработки и представления информации.

### **Оперативная коррекция периода канала**

Периодом канала можно управлять при работе системы в реальном времени. Для этого существуют следующие способы:

другим каналом типа **OUTPUT** с подтипом **КАНАЛ** и дополнением к подтипу **установить**;

с помощью форм управления;

из программ;

по сети;

по последовательному интерфейсу.

Во всех этих способах следует указать управляемый канал, а в качестве атрибута - период (**FRQ**).

### **Пример**

Оперативная коррекция периода канала.

При настройке контура регулирования (подобрать оптимальные настройки регулятора) необходимо видеть на экране переходный процесс регулирования. В это время запрос значений параметра у контроллера, реализующего регулирование, надо производить значительно чаще, чем в рабочем режиме. Поэтому период канала, получающего значения

регулируемого параметра, следует задать минимальным. После завершения настройки надобность в частом опросе отпадает, и период канала можно опять изменить на рабочее значение.

### Время изменения

Значение этого атрибута соответствует времени последнего изменения реального значения канала.

Данный атрибут возвращает 3-байтовый фрагмент из 4-байтового числа секунд с 1.01.70, т.е. число секунд с 1.01.70 по модулю  $2^{24}$ .

### Атрибуты СПАД, Регистратор, Отчет тревог, Атрибуты

Данные атрибуты возвращают 1 при установке соответствующего флага в настройках канала, в противном случае их значение равно 0.

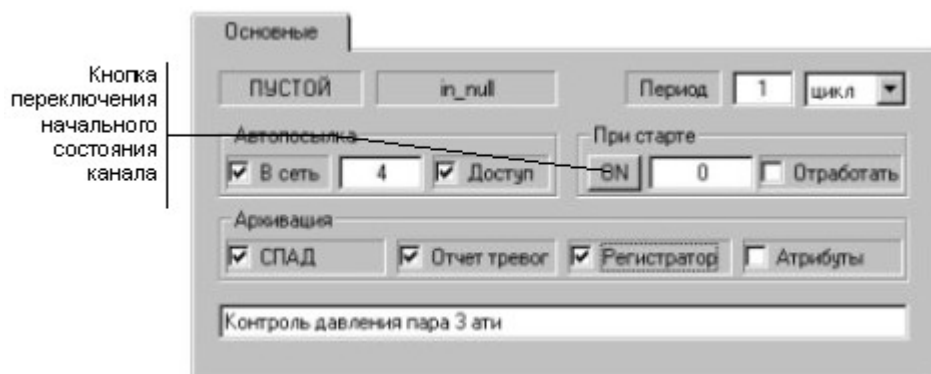
### Состояние канала

Каждый канал ТРЕЙС МОУД может находиться в одном из двух состояний – **включен** и **выключен**. Если канал включен (**ON**), то его значения пересчитываются с частотой, определяемой периодом канала. Когда канал выключен (**OFF**), значения в нем не пересчитываются и остаются неизменными.

Состоянием канала можно управлять в реальном времени. Для этого можно использовать методы, что и для управления периодом канала. Надо только в качестве управляемого атрибута указать состояние канала (**C**).

Для каждого канала можно установить начальное состояние. Это состояние, в которое канал будет переведен при запуске системы. По умолчанию начальное состояние канала – включен.

Начальное состояние канала задается в бланке **Основные** диалога **Реквизиты**. Установка начального состояния канала производится нажатием ЛК на соответствующей кнопке раздела **При старте**.



Примером использования возможности управления состоянием каналов может быть задача квитирования аварийного звукового сигнала. В качестве другого примера можно привести отключение опроса датчиков состояния оборудования, используемого только в моменты пуска или остановки технологического процесса.

## Начальное значение канала

При настройке канала можно задать его начальное значение. Это значение при запуске монитора реального времени присваивается входному значению канала.

Начальное значение, как и начальное состояние канала, задается в бланке **Основные** диалога **Реквизиты**.



Чтобы начальное значение канала после присвоения было пересчитано процедурой трансляции, а в случае выходных каналов были произведены действия, определенные подтипом и дополнением к подтипу, в этом же меню следует установить флаг **отработать**.

## Флаги архивирования

Система архивирования ТРЕЙС МОУД включает в себя три типа архивов: **локальный архив**, **отчет тревог** и **глобальный регистратор**. Они отличаются механизмом сохранения значений каналов и форматом файлов.

В **локальный архив** значения каналов записываются в бинарном виде при их изменении. В нем реализована СПАД-технология архивирования данных. Это обеспечивает быстрый доступ к архивным данным и защиту от переполнения диска.

**Отчет тревог** ведется в ASCII-формате. Алгоритм записи сообщений в этот архив отличается для каналов, обрабатывающих аналоговые данные и дискретные. В первом случае сообщения заносятся при пересечении значением канала аварийных границ. Во втором - при изменении значений указанных битов канала.

В **регистратор** сохраняются значения каналов со всех узлов проекта. Запись значений осуществляется по их изменению. Он также реализует СПАД-технология "хранилища данных". Данные в нем хранятся в бинарном формате.

Подробно задача архивирования данных в ТРЕЙС МОУД рассматривается в разделе **Архивирование**, где описываются форматы архивов, средства управления архивированием и настройки архивов.

Чтобы значение канала записывалось в любой из этих архивов, необходимо установить для него соответствующий флаг:

флаг **СПАД**;

флаг **ТРЕВОГИ**;

## флаг РЕГИСТРАТОР.

Перечисленные флаги устанавливаются для канала в разделе **Архивация** бланка **Основные** диалога **Реквизиты**. Установка/снятие флагов осуществляется нажатием ЛК в соответствующих областях бланка (см. рисунок).



Чтобы в архивах сохранялись также изменения всех остальных атрибутов канала, надо установить флаг **Атрибуты**.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 4. [Редактор базы каналов Trace Mode 5.](#)**  
Справочная система Trace Mode 5, раздел: «Редактор базы каналов», глава:  
«Редактирование базы каналов», «Объект базы каналов», «Редактирование каналов  
объекта», «Первичная и выходная обработка», «Отладка алгоритмов».

## ЛЕКЦИЯ 5 ЗАДАЧИ SCADA-СИСТЕМ

1. Управление аппаратурой, визуализация, мониторинг

2. Атрибуты канала

- интервалы и границы;
- тенденция изменения значения канала;
- достоверность значения канала;
- размерность;
- атрибуты первичной обработки;
- кодировка сигналов;
- комментарий;
- события;
- подключения.

### 1. Управление аппаратурой, визуализация, мониторинг

Первое, и самое важное обстоятельство - SCADA пакеты, в отличие от большинства программ, непосредственно связаны с процессами, происходящими на предприятии, поскольку через систему серверов ввода-вывода подключаются к разнообразнейшей аппаратуре, управляющей и контролирующей производственный процесс.

Основная роль человеко-машинного интерфейса - обеспечить взаимодействие пользователя с устройствами ввода/вывода и через них с контролируемым оборудованием.

SCADA - обеспечивают взаимодействие не только с устройствами ввода/вывода, но и с другими источниками информации типа баз данных, Windows-программ и внешних компьютерных систем.

Вся информация с датчиков, управляющих механизмов и промышленных контроллеров в реальном времени поступает в SCADA - системы. Для обеспечения простоты подключения этих устройств в них включены сотни готовых серверов ввода-вывода, ориентированных на конкретные типы оборудования. Кроме этого, наличие стандартных интерфейсов, таких как OPC, DDE, ActiveX, TCP/IP и других позволяет применять серверы, разработанные другими компаниями, что гарантирует открытость решения.

Поступающие в SCADA - систему данные не только сохраняются во внутренней базе данных, но и могут обрабатываться согласно определенным инженерами алгоритмам, что дает возможность реализации систем автоматического управления, то есть управления без участия оператора. Конечно, наиболее ответственные задачи, такие как система противоаварийной защиты, должны, по крайней мере, дублироваться на уровне контроллера, но большинство супервизорных алгоритмов может быть реализовано в системе SCADA.

## *Сигналы тревоги*

Наряду с алгоритмами обработки в современной SCADA - системе предусмотрена возможность автоматической генерации сигналов тревоги в соответствии с заданными инженерами или технологами критериями. Возникающие сигналы тревоги могут отображаться на экране, записываться в журнал и быть доступны одновременно для нескольких рабочих мест оператора. Подтверждение этих сигналов, осуществляется операторами, имеющими соответствующие права доступа к системе. С каждой «тревогой» можно связать определенное действие, которое будет выполняться при появлении этой «тревоги» (например, запуск звукового файла).

Средством информирования оператора о возникновении каких-либо аварийных ситуаций и неисправностей являются конфигурируемые тревоги.

Система тревог может контролировать всё: переменные, группы переменных, выражения, результаты расчетов и т.д. Например, можно выводить сообщение, когда уровень жидкости в резервуаре станет слишком высоким, когда двигатель перегреется и т.д.

Очень большое значение имеет быстрое распознавание и идентификация тревог. SCADA - пакет выводит информацию о тревогах в специализированные окна, однако самые свежие данные видны в каждом окне. Тревоги можно группировать по цвету, шрифту и порядку вывода в зависимости от приоритета, категории и времени возникновения.

Регистрация информации в основном заключается в сборе и записи определенных аналоговых и дискретных параметров контролируемого оборудования или процесса. SCADA - система не накладывает никаких ограничений на тип регистрируемых данных и предоставляет широкий выбор функций регистрации:

- события регистрируются в момент возникновения (например, тревоги, этапы процесса, сигналы датчиков и т.д.);
- регистрируются все действия оператора (типа ручного запуска процесса, аварийного останова, изменения контрольных показателей и т.д.);
- регистрируются все ошибки и события внутри системы управления (аппаратные тревоги, сведения об обмене данными, ошибки сети и т.д.).

Очень часто назначением системы автоматизации является сбор и хранение информации, как в качестве архива, так и для дальнейшего анализа. SCADA - система позволяет архивировать данные самого разного типа без каких-либо ограничений на тип и местонахождение выводного устройства. С каждым событием может быть связано действие, которое будет выполняться в момент возникновения этого события. Например, при завершении какого-либо процесса об этом можно уведомить оператора и выполнить некоторую последовательность завершающих действий.

Событие - это некоторая возникшая в системе ситуация, имеющая для системы определенное значение (например, полное заполнение резервуара или завершение какого-либо процесса и т.д.). Контролируемые события могут иметь отношение ко всему предприятию или иметь локальное значение с точки зрения операторской станции. Если SCADA - система используется в сети, то события могут обрабатываться любым компьютером.

Для того чтобы в интуитивно понятной для оператора форме отображать на экране компьютера текущее состояние производственного процесса или мнемосхемы, в SCADA - системе встроены специальные графические средства.

### ***Графические средства SCADA***

Средства визуализации - одно из базовых свойств SCADA - систем. В каждой из них существует графический объектно-ориентированный редактор с определенным набором анимационных функций. Используемая векторная графика дает возможность осуществлять широкий круг операций над выбранным объектом. Объекты могут быть простыми (линии, прямоугольники, текстовые объекты и т. д.) и сложные. Возможности агрегирования сложных объектов в разных SCADA - системах различны. Все SCADA - системы включают библиотеки стандартных графических символов, библиотеки сложных графических объектов, обладают целым рядом других стандартных возможностей.

Графики строятся на базе весьма простого набора графических объектов, а именно: прямоугольников, эллипсов, точечных изображений, отрезков, кривых, ломаных, текста, символов и труб. У каждого объекта есть некоторый, общий для всех набор свойств. Все они могут быть непосредственно связаны с параметрами контролируемого оборудования, которые будут определять поведение графических объектов.

Перемещение, вращение, изменение размеров, цвета, заполнения, видимости на экране и т.д. любого объекта могут выполняться в зависимости от реальных производственных условий, а изменение параметров процесса может быть выполнено путем изменения параметров объекта.

На действия оператора могут реагировать все объекты, поэтому операторский интерфейс может быть сделан настолько простым, интуитивно понятным и гибким, насколько это возможно.

В комплект поставки SCADA - системы входят библиотеки образов, содержащие наиболее часто используемые графические изображения типа насосов, резервуаров, вентилях, двигателей и т.д. Их применение существенно расширяет возможность пользовательских экранов.

### ***Генерация отчетов***

Отчет SCADA - системы - это документ, отражающий некоторые производственные показатели и выдаваемый периодически, по запросу либо

при возникновении какого-либо события, например, при изменении состояния какой-либо переменной, в момент запуска SCADA - системы или в указанное время дня.

Отчеты могут генерироваться в любом удобном для пользователя формате. В него может входить форматированный текст, оперативная и накапливаемая информация и даже результаты математических вычислений. Кроме того, отчеты могут содержать и некоторые команды: замена производственных параметров, загрузка инструкций, выполнение диагностики, смена составов смесей и т.д. Отчеты могут выводиться на экран, распечатываться, а также сохраняться на диске для последующей распечатки или просмотра. Отчет можно обрабатывать средствами любого текстового редактора. Их можно автоматически сохранять в SQL-базах и других ODBC-совместимых базах данных.

### ***Защита от несанкционированного доступа***

Практически во всех системах определенный набор действий должен выполняться только уполномоченным персоналом.

Используемый человеко-машинный интерфейс должен обеспечивать определенный уровень защиты во избежание случайного или преднамеренного исполнения запрещенных операций.

Защита от несанкционированного доступа интегрирована во все интерфейсные элементы SCADA – системы, гарантируя полную безопасность исполнительной системы.

Системы защиты от несанкционированного доступа в большинстве SCADA – систем, реализованы на базе парольной системы и позволяют организовывать в системе группы пользователей с различными правами во время работы с системой. Каждому пользователю назначаются свои регистрационные имя и пароль, которые он должен указывать для получения доступа к различным компонентам системы. Права пользователя определяются предоставлением ему возможности доступа к тем или иным частям системы. Даже имея право доступа, к какой-либо части для выполнения тех или иных действий, пользователь должен обладать соответствующим уровнем привилегий.

Каждый графический объект, окно, тренд, отчет и т.д. можно привязать к определенной части системы и определить необходимый для их просмотра или использования уровень привилегий.

Поскольку пользователь может работать на любом компьютере сети, то предоставление прав доступа контролируется сервером, а не клиентом (еще одно дополнительное средство защиты от несанкционированного доступа в глобальных сетях).

### ***Резервирование***



В промышленных системах автоматизации и прочих ответственных приложениях отказы оборудования приводят к замедлению производства и иногда к возникновению потенциально опасных ситуаций.

Устранять отказы в системе без потери ее функциональных возможностей и производительности позволяет реализация функций резервирования.

Благодаря дублированию устройств ввода/вывода, SCADA - система поддерживает конфигурации с полным резервированием. Определив одно устройство как основное, а другое как резервное, SCADA - система в случае отказа будет автоматически переключаться с одного на другое. Благодаря способности SCADA - системы записывать изменения контрольных параметров как в основное, так и резервное устройство, даже те устройства ввода/вывода, которые проектировались без учета этой возможности, могут теперь использоваться в системах с резервированием.

Резервирование в SCADA - системе тесно связано с системой тревог. В случае сбоя SCADA - система уведомит оператора об отказе конкретного устройства и сообщит, какое резервное оборудование было включено в работу.

## 2. Атрибуты канала

- интервалы и границы;
- тенденция изменения значения канала;
- достоверность значения канала;
- размерность;
- атрибуты первичной обработки;
- кодировка сигналов;
- комментарий;
- события;
- подключения.

### **Интервалы и границы**

[Границы](#)

[Интервал](#)

[Гистерезис](#)

[Контроль шкалы](#)

В любом технологическом процессе есть параметры, по значению которых определяется состояние отдельных технологических узлов, стадий, участков, аппаратов. Выход значений этих параметров за заданные границы указывает на предаварийное или аварийное состояние процесса. При возникновении такой ситуации система управления должна выполнить действия, направленные на предотвращение возникновения или

развития аварии (включение сигнализации, выдача рекомендации оператору или блокировка).

Чтобы обеспечить обработку аварийных ситуаций, каналы, работающие с аналоговыми переменными (вид представления **F**), имеют четыре **аварийные границы** и две **границы шкалы**.

## Границы

**Границы шкалы** указывают возможный диапазон изменения контролируемого параметра. Например, если датчик позволяет измерять давление в диапазоне от 0 до 10 кгс/см<sup>2</sup>, то его показания, лежащие вне данного диапазона, являются заведомо недостоверными. Если задать для канала границы шкалы, то при выходе за них его реального значения может автоматически формироваться признак недостоверности данных. Эта информация может быть доведена до оператора и зафиксирована в архивах.

Четыре аварийных границы делятся на две верхние и две нижние. Внутренние (верхняя и нижняя) границы - предаварийные, внешние - аварийные. Значения границ задаются в разделе **Границы** бланка **Границы и обработка** диалога **Реквизиты**.

В областях **ВГ\_1**, **НГ\_1** вводятся значения верхней и нижней внешних границ; **ВГ\_0**, **НГ\_0** - значения верхней и нижней внутренних границ. Верхняя и нижняя границы шкалы задаются в областях **Верхний предел** и **Нижний предел** соответственно.

Значения границ шкалы и аварийных границ доступны для изменения в реальном времени. Это можно реализовать с помощью форм отображения, программ, по сети или последовательному интерфейсу из удаленного узла.

## Интервал

Аварийные границы и шкала разбивают диапазон изменения значения канала на 7 интервалов. На рисунке приведено графическое представление границ и интервалов значений канала.

Нижний предел	НГ_1	НГ_0		ВГ_0	ВГ_1	Верхний предел
6	4	2	0	1	3	5
Диапазон изменения значения канала				Номер интервала		

В случае, когда задано не менее двух ненулевых значений границ и полный перечень заданных значений границ корректен, МРВ на каждом цикле пересчета данных

определяет номер интервала, в котором находится текущее значение канала, и формирует значение специальной переменной, являющейся атрибутом канала. Эта переменная называется **интервал** и обозначается **P**.

## Пример

Обработка аварийной ситуации.

Использование аварийных границ и интервала.

Рассмотрим решение следующей задачи: при понижении давления в котле ниже предаварийной границы (НГ\_0) надо записать в отчет тревог сообщение "КОТЕЛ\_1 предаварийное состояние" и проиграть предупреждающий звуковой файл.

Для решения этой задачи потребуются два канала. Настроим один из них на прием данных от датчика давления и зададим ему имя ДАВЛЕНИЕ. Для этого канала в диалоге **Реквизиты** установим флаг сохранения в отчет тревог и, исходя из технологических требований, зададим значение границы НГ\_0. Далее поставим для канала давления флаг сохранения в отчет тревог и в бланке **Сообщения в отчет тревог** введем требуемое сообщение для записи в отчет тревог.

Второй канал должен иметь тип OUTPUT, подтип СИСТЕМНЫЙ и дополнение к подтипу **звуковой файл**. Имя этому каналу дадим ЗВУК. Далее создадим программу, содержащую два аргумента. Эта программа должна при отличии первого аргумента от 0 формировать значение второго аргумента равным 1 (номер звукового файла, содержащего вой сирены), а в противном случае - 0. Установим ссылку на эту программу из процедуры УПРАВЛЕНИЕ канала ЗВУК. В качестве первого аргумента будем использовать значение интервала канала ДАВЛЕНИЕ, а в качестве второго - реальное значение канала ЗВУК.

Тогда при переходе реального значения канала, измеряющего давление, через границу НГ\_0 аппаратное значение канала, управляющего звуковой платой, будет равно 1. Файл с записанным звуковым предупреждением должен находиться в директории проекта и иметь имя 1.wav.

## Гистерезис

В бланке **Границы и обработка** можно задать еще один параметр. Он влияет на отслеживание перехода значения канала через аварийные границы и называется **гистерезис**.

Введение гистерезиса позволяет убрать ненужный поток сообщений в отчет тревог при небольших колебаниях контролируемого параметра вблизи значения одной из границ. Величина этого параметра задается в области **Гистерезис**.

Переход реального значения канала через границы в сторону развития аварийной ситуации (увеличение номера интервала) фиксируется по заданным для них значениям. При обратном изменении значения канала его границы корректируются в соответствующем направлении на величину гистерезиса. По полученным таким образом значениям границ вычисляется номер интервала.

## Контроль шкалы

Для канала, работающего с аналоговыми переменными, можно установить флаг **Контроль шкалы**. Он имеет разное назначение в зависимости от типа канала. При его наличии у каналов типа INPUT устанавливается признак недостоверности, если реальное значение выходит за границы шкалы. Если тип канала OUTPUT, то флаг контроля шкалы задает ограничение изменения его реального значения рамками границ шкалы. При попытке присвоить каналу значение, выходящее за шкалу, оно будет обрезано до значения соответствующей границы.

#### **Тенденция изменения значения канала**

Для анализа технологических ситуаций и принятия решения часто необходимо знать не только значение параметров, но и то, как они изменяются в настоящий момент. В ТРЕЙС МОУД каждый канал имеет специальный атрибут, характеризующий тенденцию его изменения. **Тенденция (D)** оценивает изменение реального значения канала на текущем цикле по отношению к предыдущему.

В зависимости от вида представления канала формирование величины этого атрибута осуществляется разными способами. Для каналов с видом представления **F** он принимает следующие значения:

- 0** - значение не изменилось;
- 1** - значение уменьшилось;
- 2** - значение увеличилось.

У каналов с видом представления **H** атрибут **тенденция** представлен целым 16-битным числом. Каждый бит этого числа является индикатором изменения соответствующего бита реального значения. Если на текущем такте пересчета значение бита изменилось, то соответствующий бит тенденции устанавливается равным 1, в противном случае - 0.

#### **Достоверность значений канала**

##### [Признак аппаратной достоверности](#)

##### [Признак программной достоверности](#)

В работе систем управления могут возникать ситуации, когда получаемые с объекта данные не могут быть однозначно приняты для управления. Примером таких ситуаций может быть неисправность датчика или обрыв линии связи. В этих случаях данные в системе будут присутствовать. Однако в первом случае передаются неверные значения контролируемого параметра, а во втором - значения параметров просто не обновляются.

Для индикации подобных ситуаций каждый канал имеет два флага достоверности его значений. Установка любого из этих флагов в 1 означает, что значения канала недостоверны. Эти признаки достоверности имеют названия:

- аппаратная достоверность,
- программная достоверность.

Признаки достоверности собраны в одном атрибуте канала, который называется **Достоверность**. Признак аппаратной достоверности формирует 0-й бит этого атрибута, а программной - 1-й бит.

### Признак аппаратной достоверности

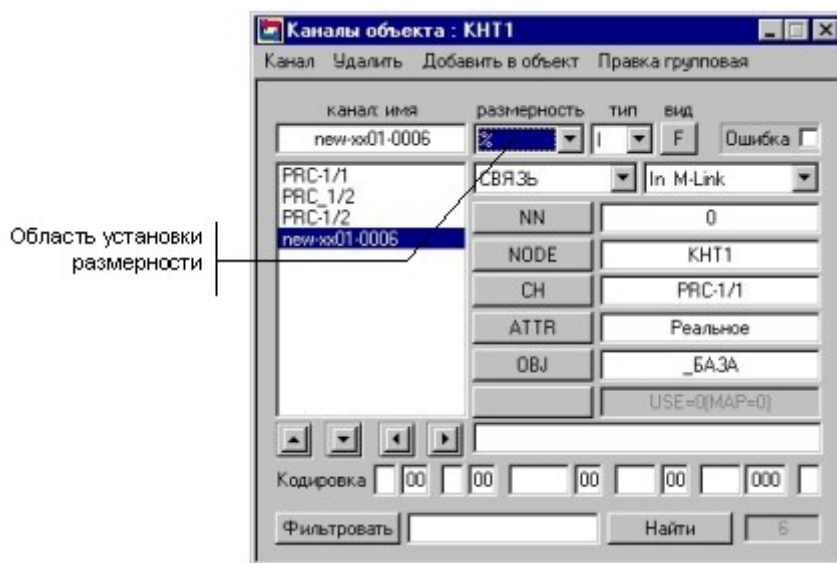
Этот признак может формироваться только для каналов, связанных с аппаратурой ввода/вывода в случае сбоя при обмене данными.

### Признак программной достоверности

Этот признак может быть сформирован программно из любой пользовательской программы. Для установки этого признака надо в атрибут **Недостоверность** послать значение 1. Формирование из программ аппаратного признака достоверности недоступно.

### Размерность

Каждому каналу можно присвоить размерность, в которой представлено его реальное значение. Установка размерности осуществляется в диалоге **Каналы объекта**. Для этого используется специальное поле данного диалога. Ниже на рисунке показан вид этого диалога, на котором обозначено поле задания размерности.



Размерность задается выбором из списка, который хранится в файле **razmer.ini** в директории **INI**. Если требуемая размерность в списке отсутствует, то ее можно добавить, отредактировав указанный файл. Текст размерности не может содержать более 8 символов, а количество строк в списке не должно превышать 255.

Установленная размерность значения канала недоступна для изменения в реальном времени. Она может выводиться на экран с помощью текстовых форм отображения.

### Атрибуты первичной обработки

[Каналы для обработки аналоговых сигналов](#)

[Каналы для обработки дискретных сигналов](#)

Каналы с различным видом представления имеют разный набор встроенных операций первичной обработки данных. Следовательно, набор атрибутов, настраивающих обработку, у них отличается.

Если встроенных методов обработки недостаточно, то дополнительную обработку можно оформить в виде программы на языке функциональных блоков и вызвать ее из процедуры трансляции.

### **Каналы для обработки аналоговых сигналов**

Атрибуты первичной обработки каналов с видом представления **F** настраиваются в бланке **Границы и обработка** диалога **Реквизиты**. Они имеют следующие методы первичной обработки:

- умножение;
- смещение;
- экспоненциальное сглаживание;
- апертура (зона нечувствительности);
- фильтрация пиков;
- контроль шкалы

Их подробное описание приведено в разделе, посвященном редактору базы каналов.

### **Каналы для обработки дискретных сигналов**

Для каналов ТРЕЙС МОУД с видом представления **N** реализованы следующие методы первичной обработки:

- инверсия;
- предустановка;
- контроль сочетаемости.

Они настраиваются в бланке **Маски и эмуляция** диалога **Реквизиты**. Для этого используются три атрибута и три флага, включающие соответствующие методы.

### **Кодировка сигналов**

[Структура кода](#)

[Варианты кодирования](#)

[Настройка кодировки канала](#)

[Описание кода](#)

В ТРЕЙС МОУД для каждого канала можно задать индивидуальный код, который определяет его привязку к технологической схеме. Например, в кодировке могут указываться цех, участок, аппарат, устройство, тип сигнала.

Можно использовать одну из трех схем кодировок: АКС, ККС и ПРИМА. Для получения информации о первых двух используйте специальную литературу. Далее будет описана кодировка ПРИМА.

С помощью кодировки можно построить иерархию объектов, которая позволяет легко ориентироваться в сложных проектах. Это облегчает для проектировщика выполнение декомпозиции задачи.

### Структура кода

Код имеет фиксированную структуру. Для описания любого сигнала используются 11 полей. Из них шесть текстовых и пять числовых. Структура кода выглядит следующим образом:

Axx.Axx.AAAxx.AAxx.AAxxx.A

где

A - одна позиция символьного поля;

x - одна позиция числового поля.

Каждая пара полей (текстовое и числовое) содержит идентификатор в рамках соответствующего уровня иерархии и номер объекта с этим идентификатором. Настройка полей хранится в отдельных файлах и доступна для редактирования. Эти файлы имеют имена **code0.cfg, ..., code5.cfg**. Они находятся в поддиректории **INI** рабочей директории. Там же хранятся файлы описания кодировки АКС (**acode0.cfg, ..., acode3.cfg**).

### Варианты кодирования

Рассмотрим, как могут кодироваться сигналы в рамках проекта ТРЕЙС МОУД.

Например, первая пара (одна позиция символьная и две числовые) может характеризовать административное деление предприятия:

Значение	Код
Цех	Ц
Служба	С
Подразделение	П
...	...

Таким образом, значение первой пары **Ц05** может означать, что сигнал относится к цеху с номером 5.

Вторая пара (**Axx**) содержит одну символьную позицию и две числовые. На этом уровне может кодироваться разделение на стадии производства. В таблице приведен пример такого кодирования:

Значение	Код
Отделение	О
Участок	У
Энергоблок	Э
...	...

Значение этой пары **У03** может означать, что сигнал относится к участку с номером 3.

Третья пара в коде (**АААхх**) содержит три символьные позиции и две позиции числовые. Этот уровень может использоваться для кодирования конкретного элемента технологического оборудования, к которому относится сигнал или команда. Пример кодирования элементов оборудования показан в следующей таблице:

Значение	Код
Теплообменник кожухотрубный	ТПК
Теплообменник пластинчатый	ТПП
Колонна	КЛН
Емкость горизонтальная	ЕМГ
...	...

Значение данной пары **ЕМГ15** может означать, что сигнал относится к датчику, установленному на емкости с номером 15.

Следующий уровень кодирования содержит две символьные позиции и две позиции числовые (**ААхх**). На нем может кодироваться устройство, контур управления или комплексного измерения. Примерами элементов этого уровня кодирования могут быть:

Значение	Код
Задвижка	ЗД
Двигатель	ДВ
Контур регулирования	КР
Шаровой кран	ШК
...	...

Значение этой пары **ЗД23** может означать, что данный сигнал относится к задвижке с номером 23.

Последняя пара в кодировке содержит две символьные позиции и три позиции числовые (**ААххх**). Этот уровень может использоваться для кодирования характеристик контролируемого или управляемого параметра. Примерами таких кодировок могут быть:

Значение	Код
Давление	ДД
Уровень	УР
Положение регулирующего клапана	УК
Сигнал с концевого выключателя	КВ



Значение	Код
Сигнал на открытие	УО
...	...

Значение данной пары ДД315 может кодировать сигнал измерения давления с номером 315.

Последнее поле в кодировке позволяет уточнить назначение сигнала. Оно содержит только одну символьную позицию (А). Примерами элементов этого уровня кодирования могут быть:

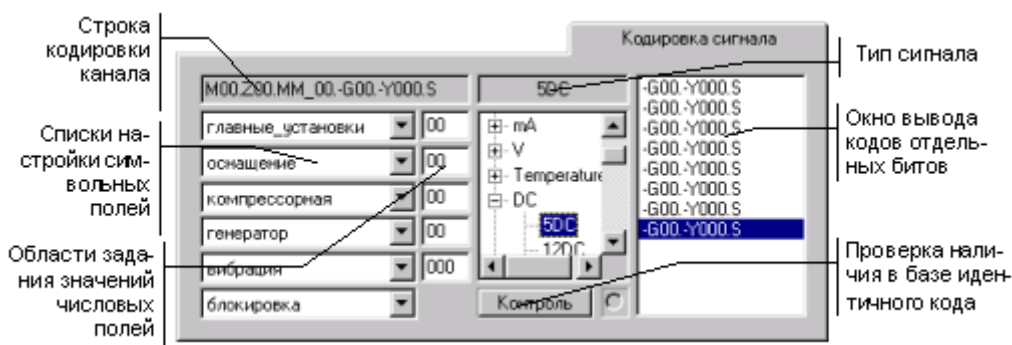
Значение	Код
Блокировка	
Измерение	
Регулирование	
Уставка	
...	...

Описанная система кодирования не является жестко фиксированной и может быть настроена произвольным образом. Неизменными являются только количество и формат полей кода.

Кроме отображаемых полей кодировки, в нее включено еще одно поле. В него записывается идентификатор типа сигнала.

### Настройка кодировки канала

Кодировка канала задается в бланке **Кодирование сигнала** диалога **Реквизиты**. Его вид показан на следующем рисунке.



Здесь имеются шесть списков для формирования символьных полей кода и пять областей для ввода числовых полей. При этом первой паре полей в кодировке соответствуют верхний список и верхнее поле.

Запись кодировки отличается для каналов с разным видом представления. Каналы, работающие с аналоговыми переменными, имеют только один код. Для каналов с видом представления **Н** количество кодов определяется числом используемых бит. Причем коды, определенные для битов одного канала, имеют одинаковую начальную часть. Они отличаются только последними пятью полями. Эти поля для каждого бита канала выводятся в специальном окне бланка **Кодировка сигнала**.

При открытии вкладки **Кодировка сигнала** в поля кодировки автоматически проставляются пробелы. Если необходимо выводить комментарий канала вместо кодировки (в отчет тревог или на графическую консоль), надо в диалоге **Каналы объекта** для этого канала в первые два поля кодировки ввести "\_".

## Описание кода

Описание кодов хранится в текстовый файлах **code0.cfg... code5.cfg** в поддиректории **INI** рабочей директории. Они содержат описания текстовых полей кодировки. Описание включает в себя тексты и соответствующие им коды. Файлы описаний имеют следующий формат:

```
<n+1>  
  
<cod_1> <name_1>  
  
<cod_2> <name_2>  
  
.....  
  
<cod_n> <name_n>  
  
##
```

где

**n** - число значений данного поля;

**cod\_i** - код i-го значения;

**name\_i** - название i-го значения.

## Комментарий

Для каждого канала можно задать комментарий: текстовую строку длиной до 38 символов. Текст комментария доступен в реальном времени, если канал не имеет кодировки. В этом случае в поля вывода кодировки подставляются первые 21 символ комментария.

Эта информация может выводиться на экраны мониторов, вставляться в отчеты и записываться в отчет тревог.

## События

Этот атрибут канала является комплексным. Он содержит информацию о номере интервала, достоверности и изменении реального значения канала.

Младший бит этого атрибута является индикатором изменения реального значения канала. Он устанавливается равным 1 при изменении реального значения канала и сбрасывается в 0 при посылке в атрибут **события** любого значения.

В биты со второго по четвертый (считая с 1) атрибута **события** записывается текущее значение номера интервала.

Пятый (считая с 1) бит этого атрибута является индикатором достоверности. В него считывается значение признака аппаратной недостоверности при послылке в атрибут **события** любого значения.

Бит 6 - копия флага ОТРАБОТАТЬ, который взводится в соответствии с условиями обработки канала и сбрасывается после успешного исполнения функции канала.

В седьмой и восьмой биты атрибута **события** записывается значение 3 при изменении номера интервала или при появлении признака аппаратной недостоверности.

При послылке в атрибут СОБЫТИЯ любого значения первый бит данного атрибута сбрасывается в 0, в пятый – считывается текущее значение признака аппаратной недостоверности, а в биты 7 и 8 записываются значения двух младших битов введенного числа соответственно. Остальные биты атрибута при этом не изменяются.

### **Подключение**

Каждый канал имеет свой источник или приемник данных. Для настройки канала на источник/приемник используются его тип, подтип, дополнение к подтипу и настройки.

В резервированных системах управление ведет основной узел, резервный только копирует значения каналов основного узла по сети. В этом случае каналы резервного узла, связанные с внешними устройствами, надо отключить от их источников/приемников.

Для управления подключением предусмотрен специальный атрибут канала – **подключение**. При его равенстве 1 – канал отключается, а при 0 – подключается. По умолчанию значение подключения для всех каналов равно 0.

Атрибут **подключение**, равный 1, в каналах типа **INPUT** отключает всю первичную обработку канала (тракт от входа до реального значения)

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 5. [Языки разработки алгоритмов Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Общие положения», «Язык функциональных блоков»:**

- создание и атрибуты FVD-программ;
- вызов программ;
- основные понятия;
- редактирование программ;
- описание функциональных блоков (логические функции, арифметические функции, тригонометрические функции, алгебраические функции, функции сравнения).

## ЛЕКЦИЯ 5

# ЭТАПЫ ЖИЗНЕННОГО ЦИКЛА ПРОМЫШЛЕННЫХ ИЗДЕЛИЙ

### 1. Этапы жизненного цикла сложных промышленных изделий

в концепции CALS технологий.

2. Проектирование.

3. Языки программирования алгоритмов в Trace Mode 5  
- язык функциональных блоков (Техно FBD);  
- язык инструкций (Техно IL).

Вестник МГТУ. Серия Приборостроение. – 2002. №1. Изд. МГТУ им.

Н.Э.Баумана.

Е.Г.Юдин, П.К.Кузьмик, И.П.Норенков

### 1. Этапы жизненного цикла сложных промышленных изделий

#### в концепции CALS технологий

*Совокупность методов и средств информационной поддержки различных этапах жизненного цикла наукоемких машиностроительных изделий рассматривается в концепции CALS технологий. Реализация CALS технологий достигается путем формирования единого информационного пространства, объединяющего автоматизированные системы конструирования (CAD), инженерного анализа (CAE), технологической подготовки производства (CAM), управления структурой изделия (PDM), управления документами и проектами (EDM), планирования и управления производством (MRP) и ресурсами предприятия (ERP). В каждой из систем создается специфическая информация об изделии. Совокупность множества моделей формирует интегрированное целостное описание изделия, которое сопровождает его на всем протяжении жизненного цикла.*

В последнее десятилетие на многих предприятиях страны ведется активное внедрение информационных систем, предназначенных для комплексного решения задач автоматизации проектирования (CAD), инженерного анализа (CAE) и технологической подготовки производства (CAM).

Мировая практика показывает, что использование таких интегрированных систем позволяет в 2-3 раза сократить время производства, на 30-40% понизить стоимость изделия, на 90-95% уменьшить количество ошибок. Предприятия получают реальную возможность быстрее и гибче реагировать на запросы рынка, использовать современные машиностроительные технологии, выйти на международную сертификацию и, в конечном счете, обеспечить сбыт своей продукции. Не менее важным результатом применения информационных технологий является стабильность качества выпускаемой продукции.

Расширение международной кооперации при изготовлении сложной технической продукции приводит к необходимости использования комплексных информационных систем на всех этапах жизненного цикла изделия. В соответствии с международным стандартом ISO 9000 (управление качеством продукции) к числу этих этапов относятся:

маркетинг, поиск и изучение рынка, проектирование и/или разработка технических требований к создаваемой продукции, материально-техническое снабжение производственных процессов, подготовка и разработка технологических процессов, производство, контроль, проведение испытаний и обследование, упаковка и хранение, реализация или распространение продукта, монтаж, эксплуатация, техническая помощь в обслуживании, утилизация по завершении использования.

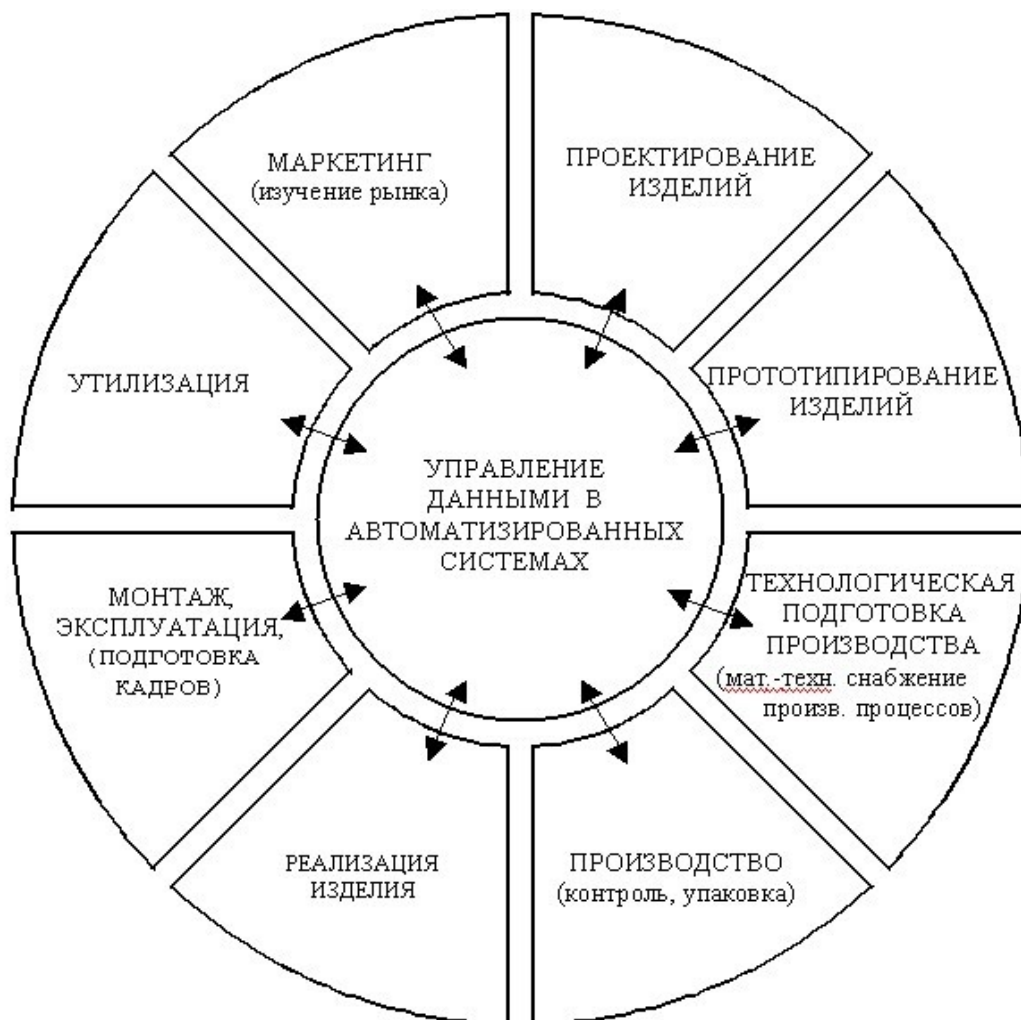


Рис. 1. Основные этапы жизненного цикла машиностроительных изделий

Совокупность методов и средств информационной поддержки жизненного цикла промышленной продукции находит свое отражение в концепции CALS технологий (Continuous Acquisition and Life-Cycle Support). В данной публикации рассматриваются вопросы инженерной составляющей жизненного цикла машиностроительных изделий (рис. 1). В практическом плане реализация CALS технологий предполагает организацию единого информационного пространства, объединяющего автоматизированные системы, предназначенные как для эффективного решения задач инженерной деятельности (конструирования (*CAD*), инженерного анализа (*CAE*), технологической подготовки производства (*CAM*), управления структурой изделия (*PDM*), управления документами и проектами (*EDM*), так и для планирования и управления производством (*MRP*) и ресурсами предприятия (*ERP*). В единый процесс вовлекается множество проектирующих и машиностроительных предприятий с удаленным доступом к информации, прямой передачей информации от компьютера к машиностроительному оборудованию и т.д. Каждому этапу жизненного цикла присущи специализированные виды обеспечения: технического, программного, информационного, математического, лингвистического,

организационного и методического. Важным аспектом информационной поддержки жизненного цикла является организация обмена данными между автоматизированными системами, участвующими в процессах проектирования, производства, эксплуатации и утилизации изделий. В каждой из систем создается своя специфическая информация об изделии. Это могут быть геометрические, расчетные модели, модели технологических процессов и т.п. Совокупность множества моделей создает интегрированное целостное описание изделия, которое сопровождает его на всем протяжении жизненного цикла.

Другими словами, речь идет не о повышении уровня автоматизации отдельных подразделений, а об организации принципиально новой информационной среды жизни изделия.

Предприятия, владеющие современными CALS технологиями, получают возможность участвовать в международной кооперации разделения труда, становятся более конкурентоспособными. Так, любое головное предприятие, отвечающее за разработку такого сложного технического изделия как самолет, автомобиль, или ракетный комплекс, как правило, занимается их сборкой, продвижением на рынок, оказывает техническую помощь в обслуживании. Производством огромного количества деталей этих изделий заняты тысячи других предприятий различных стран. В условиях жесткой конкурентной борьбы за распределение заказов побеждает то предприятие, которое, используя передовые технологии, готово предложить более выгодные условия сотрудничества, ускорить производство и обеспечить качество продукции.

Рассмотрим примеры автоматизированных систем, наиболее широко используемых на различных этапах жизненного цикла машиностроительных изделий.

## 2. Проектирование

Одним из наиболее важных этапов является этап проектирования. На этом этапе формируется геометрическая модель машиностроительного изделия или, так называемая, мастер - модель, которая будет играть определяющую роль на многих последующих этапах. На этом этапе выполняются различные виды инженерного анализа.

В автоматизированном проектировании объемное моделирование изделия является первоосновой. Здесь на помощь конструктору приходит как его собственный опыт, так и результаты работы других специалистов, воплощенные в рисунках, эскизах, чертежах, реально выполненным образцам изделий в материале, данные сканирования этих образцов и компьютерные геометрические модели ранее разработанных изделий.

Для создания объемной модели изделия конструктор может воспользоваться методами трехмерного твердотельного, поверхностного моделирования или сочетанием этих методов в адаптивных формах.

Во всех программных комплексах и системах, реализующих объемное моделирование, есть средства разработки конструкторской документации. Наряду с 3-х мерной геометрической моделью изделия, конструкторская документация представляет собой другую составляющую интегрированной компьютерной модели этого же изделия.

На сегодняшний день все существующее программное обеспечение автоматизированного конструирования принято классифицировать по функциональной полноте. По этому признаку оно делится условно на три уровня. К нижнему уровню относятся программы, реализующие 2D модели в виде чертежей и эскизов. Например, пакеты российских разработчиков БАЗИС-Конструктор 4.5 (Базис), Графика-81 (Институт проблем управления), SprutCAD (СПРУТ-Технологии), чертежно-графический редактор АРМ Graph (НИЦ АПМ), CADMECH и CADMECH LT на базе AutoCAD и AutoCAD LT2000 (Интермех) T-Flex CAD LT (Топ Системы), КОМПАС 5 (Аскон) и др.

На среднем - располагаются программные комплексы, которые позволяют создать 3-х мерную геометрическую модель сравнительно несложного изделия, в основном, методом твердотельного моделирования. К числу этих программных комплексов можно отнести: AutoCAD 2000 и AMD (AutoDesk), Solid Works (Solid Works), Solid Edge (Unigraphics

Solutions), PowerSHAPE (Delcam plc), Prelude Design (Matra Datavision), MicroStation (Bentley Systems), ГеММа-3D (ГеММа), T-Flex CAD 3D (Топ Системы), bCAD (ПроПро Группа), CREDO (НИЦ АСК), OceanCAD и др. Программные системы сквозного проектирования и производства расположены на верхнем уровне. Среди них можно выделить: CATIA5 (Dassault Systemes), EUCLID3 (EADS Matra Datavision), UNIGRAPHICS (Unigraphics Solutions), Pro/ENGINEER и CADD5 (PTC).

Обобщенная схема программного обеспечения конструирования машиностроительных изделий показана на рис. 2. Здесь же приводятся некоторые функции, которые часто реализованы в программных средах этого этапа.

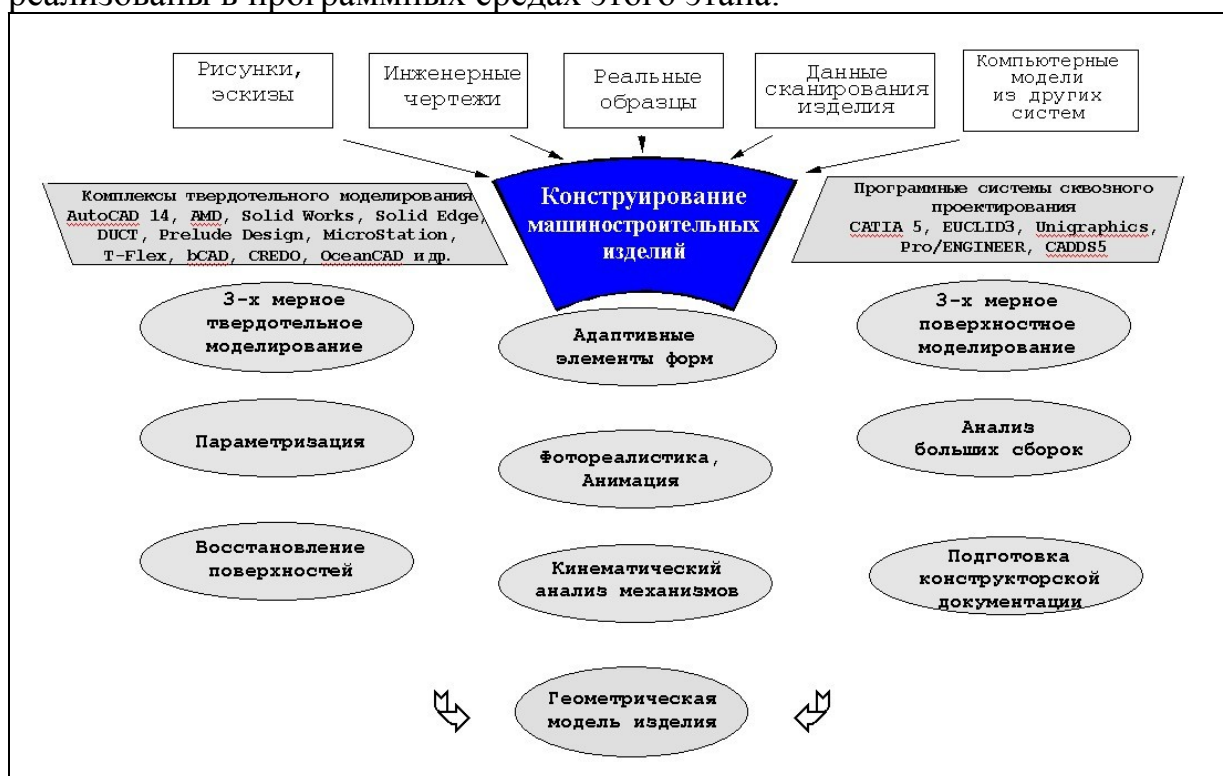


Рис. 2. Основные методы преобразования информации на этапе конструирования

С точки зрения CALS технологий программное обеспечение автоматизированного конструирования должно удовлетворять не только требованию функциональной полноты. При выборе и установке той или иной конфигурации ПО важно учитывать специфику моделей и задач, решаемых на каждом рабочем месте. В этом случае вместо одного пакета множеством универсальных функций должны устанавливаться строго специализированные пакеты программ, разработанные в соответствии с этими задачами.

Другим важным аспектом является организация коллективной работы специалистов в составе рабочих групп в интерактивном режиме (дизайнеров, конструкторов, прочнистов, технологов и т.д.). На смену последовательному сквозному проектированию приходит параллельное проектирование и технологическая подготовка производства, так как благодаря такой организации труда достигается наивысшая производительность и существенно сокращается время разработки изделия. В этих условиях вопросы организации обмена информацией становятся актуальными. Известно, что обмен без потерь информации достигается при помощи единой базы данных для различных подсистем. Этим отличаются комплексные системы сквозного проектирования и подготовки производства, где качество обмена информацией обеспечивается разработчиками этих систем. Иногда практикуется организация прямой передачи данных между пакетами различных фирмы, когда разработчики этих пакетов сочли необходимым



обеспечить прямой обмен. В большинстве случаев на рабочих местах устанавливаются программные среды различных фирм-разработчиков, и организация обмена информацией ложится на самих пользователей. Поэтому важно чтобы для этих программ были разработаны соответствующие интерфейсы с необходимой полнотой реализации форматов.

Более подробному изложению особенностей автоматизированного проектирования в машиностроении будет посвящена отдельная публикация.

### 3. Языки программирования алгоритмов

Каналы ТРЕЙС МОУД имеют встроенные функции первичной и выходной обработки. Однако этих функций недостаточно для выполнения сложных алгоритмов обработки данных, регулирования и управления. Такие алгоритмы разрабатываются в виде отдельных программ и могут вызываться процедурами канала. Для разработки таких программ в ТРЕЙС МОУД предусмотрены следующие языки:

язык функциональных блоков (Техно FBD) со встроенным языком Техно LD (Ladder Diagram);

язык инструкций (Техно IL).

Эти языки реализуют стандарт МЭК-1131. Кроме функций, описанных в этом стандарте, в них встроено большое количество дополнительных функций.

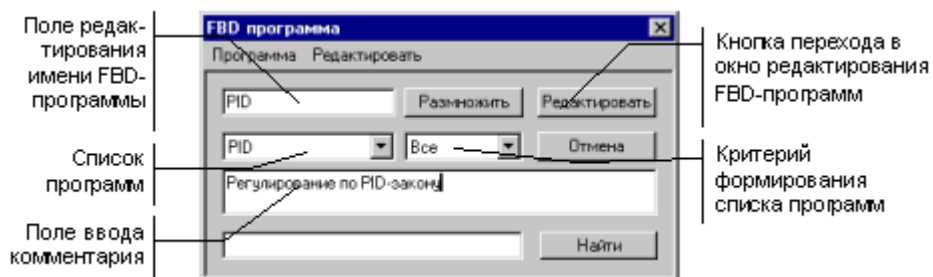
#### **Язык функциональных блоков (Техно FBD)**

Язык функциональных блоков ТРЕЙС МОУД является языком визуального программирования. Программа в нем разрабатывается размещением функциональных блоков с заданными функциями в поле редактирования, настройкой их входов и выходов, и связи их между собой в диаграмму, реализующую требуемую функцию.

#### **Создание и редактирование FBD-программ**

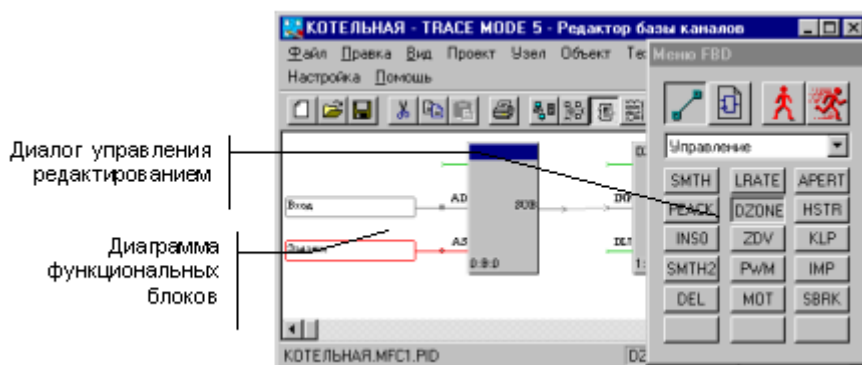
Для разработки FBD-программ в редакторе базы каналов предусмотрены два окна – **FBD программы** и **LD программы**. Для входа в первое из них нужно выполнить команду **FBD программы** меню **Окна** или нажать сочетание клавиш **ALT-3** или нажать ЛК на иконке

панели инструментов. Для входа во второе окно нужно выполнить команду **LD программы** меню **Окна** или нажать сочетание клавиш **ALT-6**. В обоих случаях на экране появляется следующий диалог:





В этом диалоге выбирается программа для редактирования. Здесь можно изменить ее имя. В этом же диалоге можно создать новую и удалить существующую программу, сохранить ее в файл или вставить из файла. После нажатия ЛК на кнопке **Редактировать** этого диалога осуществляется переход в выбранное окно редактирования FBD программ. Оба окна редактирования имеют вид, показанный на следующем рисунке:



В рабочем поле этого окна выводится диалог управления редактированием и диаграмма из функциональных блоков. В рабочем поле этого окна выводится диалог управления редактированием и диаграмма из функциональных блоков.

### Тиражирование FBD-программы

По умолчанию FBD-программа создается для одного узла. Чтобы использовать созданную программу на всех узлах проекта, нужно выполнить команду **Размножить** диалога **FBD-программы**. Полученные с помощью данной процедуры копии программы могут в дальнейшем редактироваться независимо друг от друга.

### Вызов FBD-программ

FBD-программы вызываются из процедур канала: **трансляция** и **управление**. Выбор программы и связь ее аргументов с атрибутами каналов производится из соответствующих бланков диалога **Реквизиты**. В них выводится список FBD-программ, доступных для использования. После выбора программы в специальном окне выводится для настройки список ее аргументов и констант.

### Функциональный блок

[Выполняемая функция](#)

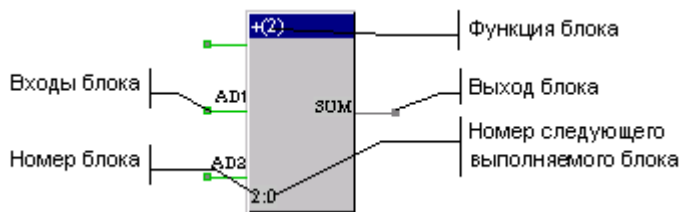
[Номер блока](#)

[Входы и выходы](#)

Элементарным звеном разработки программ на языке Техно FBD является функциональный блок.

**Блок** - это графическое изображение вызова функций. Функции могут быть либо встроенные в систему, либо разработанные пользователем на языке Техно PL или как внешние **DLL** модули.

Изображение функционального блока приведено на следующем рисунке.



Каждый функциональный блок имеет следующие атрибуты:

выполняемая функция;

номер;

входы;

выходы.

### Выполняемая функция

Наименование выполняемой функции выводится в его верхней части. Подробно доступные для использования функции описаны в разделе, посвященном описанию языков разработки алгоритмов.

### Номер блока

Номер функционального блока устанавливается автоматически при его размещении в рабочем поле. Этот номер недоступен для изменения. Он используется только для индикации очередности выполнения функциональных блоков при вызове программы.

Номер следующего выполняемого блока определяется автоматически. Он записывается за номером текущего блока через символ ":".

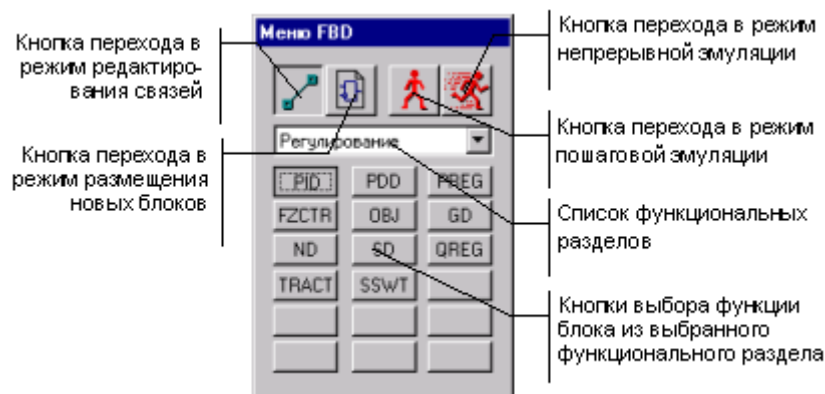
### Входы и выходы

Каждый блок в зависимости от выполняемой функции имеет определенное количество входов и выходов. Входы всегда расположены слева, а выходы - справа. При вычислении блока над переменными, связанными с его входами, осуществляются действия, определенные функцией блока. Полученные в результате значения присваиваются переменным, связанным с выходами блока.

### Редактирование FBD-программ

При входе в окно редактирования функций на экран выводится диалог **Меню FBD**. В этом диалоге осуществляется выбор режима работы и функции блока при его размещении в рабочем поле.

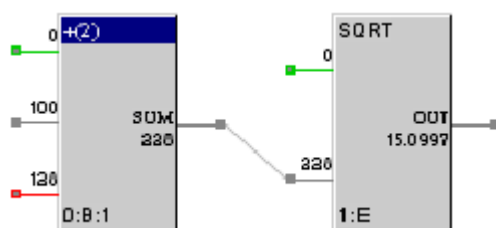
Вид этого диалога показан на следующем рисунке.



Существуют следующие режимы работы в окне редактирования FBD-программ:

- редактирование связей;
- размещение блоков;
- эмуляция работы функции.

В режиме эмуляции рядом с каждым входом и выходом всех функциональных блоков выводятся их значения.



Чтобы изменить значения входов типа **константа** или **аргумент**, следует нажать ЛК на соответствующем входе блока и в появившемся диалоге ввести требуемое значение.

Подробное описание разработки и отладки программ на языке функциональных блоков приведено в разделе, посвященном описанию языков разработки алгоритмов. Там же приводится описание всех функций, доступных для использования в этом языке.

### ***Язык инструкций (Техно IL)***

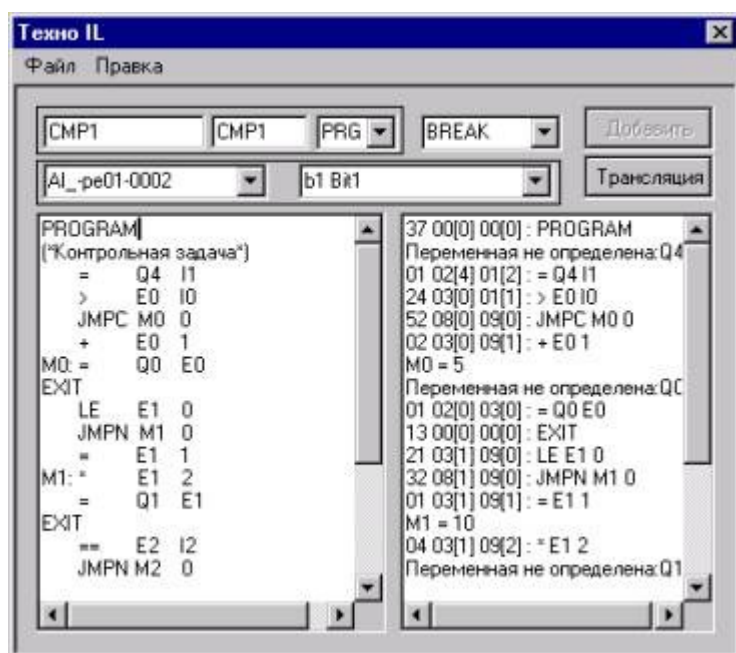
Этот язык является расширением языка IL, описанного в стандарте МЭК 1131-3. В нем реализованы все функции данного языка, а также добавлены дополнительные возможности. Язык Техно IL можно использовать для следующих двух задач:

- программирование собственных функциональных блоков для языка Техно FBD;
- разработка метапрограмм, запускаемых параллельно с пересчетом базы каналов.

IL-программы разрабатываются в текстовом виде как последовательность инструкций, содержащих команды, операнды и операторы. Тексты программ сохраняются в файлах с расширением **il** в поддиректории **ASM** рабочей директории.

### **Создание и редактирование IL-программы**

Программы на языке Техно IL разрабатываются в специальном диалоге редактора базы каналов, показанном на рисунке.



Для входа в этот диалог надо выполнить команду **Создать меню Техно IL**. Редактирование программы осуществляется непосредственным вводом текста с клавиатуры в левом окне данного диалога.

## Типы IL-программ

### [Функциональные блоки](#)

### [Метапрограммы](#)

Существует два типа IL-программ:

функциональные блоки для Техно FBD;

метапрограммы.

Тип разрабатываемой программы указывается в специальном поле диалога **Техно IL**. Здесь при разработке функционального блока следует установить значение **FB**, а для метапрограмм - **PRG**.

## Функциональные блоки

Разработанные на **Техно IL** функциональные блоки помещаются в один из трех разделов: Техно IL\_1, Техно IL\_2, Техно IL\_3. Заполнение функций этих разделов осуществляется последовательно, по мере добавления новых блоков.

## Метапрограммы

На языке **Техно IL** можно создавать программы управления и обработки данных, которые запускаются параллельно с пересчетом базы каналов. Они могут ссылаться на значения атрибутов каналов, а также вызывать в качестве подпрограмм стандартные

функциональные блоки языка Техно FBD. Максимальное количество таких программ равно 16.

### **Трансляция и подключение ПЛ-программы**

Чтобы подключить ПЛ-программу к системе, следует произвести ее трансляцию. Для этого надо нажать ЛК на кнопке **Трансляция**. Сообщения о результатах трансляции выводятся в правом окне диалога **Техно ПЛ**.

Если трансляция программы завершена успешно, то ее можно добавить в систему. Для этого надо нажать ЛК на кнопке **Добавить**. Если программа имеет тип **FB**, то в один из трех разделов Техно ПЛ\_1, Техно ПЛ\_2 или Техно ПЛ\_3 будет добавлен новый блок, имя которого совпадает с именем ПЛ-программы

Если тип программы был **PRG** (метапрограмма), то после выполнения операции добавления к базе каналов текущего узла будет подключена новая метапрограмма.

Для редактирования добавленных в систему ПЛ-программ надо выполнить команду **Редактировать** из меню **Техно ПЛ** редактора базы каналов. При этом на экран выводится диалог для выбора редактируемого функционального блока или метапрограммы.

Один и тот же исходный текст метапрограммы может быть использован в разных узлах проекта. Полученные таким образом копии программы могут в дальнейшем редактироваться независимо друг от друга.

Особенности оформления текста метапрограмм, а также дополнительные возможности языка Техно ПЛ при их разработке рассмотрены в разделе, посвященном описанию языков разработки алгоритмов.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 6. [Языки разработки алгоритмов Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»:**

**- описание функциональных блоков (функции выбора, триггеры и счётчики, генераторы).**

## ЛЕКЦИЯ 7

### ЭТАПЫ ЖИЗНЕННОГО ЦИКЛА ПРОМЫШЛЕННЫХ ИЗДЕЛИЙ

1. Инженерный анализ
2. Прототипирование
3. Технологическая подготовка производства
4. Производство
5. Разработка графического интерфейса в Trace Mode 5.
  - редактор представления данных;
  - элементы графического интерфейса.

#### 1. Инженерный анализ

В последние годы наметилась тенденция более узкой специализации фирм-разработчиков программных средств анализа. Мировыми лидерами в области разработки, поставки и сопровождения программных систем инженерного анализа (CAE) машиностроительных изделий являются MSC.Software Corporation, SAMTECH, ANSYS и некоторые другие. Во всех программных средах этих фирм используются методы конечных элементов. Выше отмечалось, что точная геометрическая модель исследуемого изделия предварительно создается на этапе конструирования в той или иной CAD системе. Для проведения какого-либо вида анализа, обычно, в этой же CAD системе на основе точной геометрической модели создается расчетная (упрощенная) модель путем удаления тех конструктивных элементов, которые не оказывают существенного влияния на результаты анализа. Расчетная модель передается в пакет анализа при помощи стандартных интерфейсов. Отдельные пакеты анализа имеют внутренние средства построения геометрической модели, с помощью которых может быть решена задача моделирования простых форм.

Основные программные разработки фирмы MSC это MSC.NASTRAN и MSC.PATRAN. Первый пакет предназначен для решения широкого спектра задач анализа линейной и нелинейной статики и динамики, устойчивости, теплопередачи, акустики, аэроупругости, оптимизации конструкции. Вторая разработка представляет собой интегрированное средство моделирования и анализа нового поколения. Одной из интересных разработок MSC, выполненных в последние годы, является пакет MSC.SuperForge, предназначенный для объемного моделирования процессов штамповки иковки. Результаты анализа могут быть использованы для проектирования оснастки и технологических процессов [ 1 ]. Попутно заметим, что, помимо американской фирмы MSC, признанными лидерами в области моделирования процессов штамповки иковки также являются американская компания SFTC (система DEFORM), французская компания TRANVALOR (система FORGE) и российская фирма Квантор-Софт (система Qform [2]).

Компанией SAMTECH (Бельгия) в сотрудничестве с Лабораторией аэрокосмических технологий Льежского университета разработана комплексная система анализа SAMSEF. В ней методом конечных элементов могут проводиться различные виды анализа, включая: статический и динамический анализ, анализ устойчивости, нелинейный температурный анализ (в том числе с учетом процесса фазового перехода или химических реакций), спектральный анализ, статический анализ циклических структур, расчет электрического поля и др. Предлагаемые программные среды используются при проектировании изделий машиностроения, судостроения, аэрокосмической и электротехнической отраслей. В ней решаются такие специфические задачи, как: нелинейный теплообмен (с переходным или стационарным режимом, включая воздействие радиации), структурная оптимизация, анализ упругих механизмов, усталостные разрушения, анализ явлений вязко пластичности и многие другие задачи. Все расчетные модули связаны с единым графическим пре- и

постпроцессором BACON, который позволяет получать графическую модель изделия из различных CAD систем. Некоторые из перечисленных выше задач анализа решаются и в пакете ANSYS.

В области разработки программных пакетов инженерного анализа значительные результаты получены российскими фирмами. К сожалению, многие пакеты не имеют стандартных интерфейсов и, поэтому, не могут использоваться комплексно с CAD программами. Приведем примеры пакетов, перечень основных задач, решаемых с их помощью, и фирм, выполнивших разработку:

- Euler - динамический анализ многокомпонентных механических систем (АвтоМеханика),
- ИСПА - расчет и анализ на прочность (АЛЕКСОФТ),
- ПОЛИГОН - конечно-элементная система для моделирования литейных процессов: гидродинамические, тепловые и усадочные процессы в 3D - постановка (ЦНИИ материалов),
- Риман - расчет и анализ напряженно-деформированного состояния конструкций, решение упругих и пластических задач, в том числе штамповки и ударных напряжений (ПроПроГруппа),
- АРМ WinMachine - комплекс программ для проектирования и расчетом деталей машин, анализа напряженно-деформированного состояния конструкций и их элементов методом конечных элементов (НТЦ АПМ),
- Диана - анализ конструкций и их элементов (НИЦ АСК),
- GasDynamics Tool - моделирование газодинамических процессов (Тульский гос. университет).

Отдельную группу программного обеспечения инженерного анализа составляют пакеты, предназначенные для анализа динамических процессов. К этой группе относятся ADAMS, MBTU, ПА-9 и др.

Пакет ADAMS (Mechanical Dynamics, Inc.) может использоваться для динамического и кинематического анализа сложных механических схем механизмов, статического и модального анализа. При помощи этого пакета могут решаться задачи, например, стыковки космического аппарата, динамики полета и посадки и т.п. Двусторонняя связь с конечно-элементными пакетами (ANSYS, MSC.NASTRAN, ABAQUS, I-DEAS) позволяет встраивать неограниченное число конечно-элементных моделей в механизм для учета влияния деформируемости на поведение системы. В ADAMS обеспечен обмен информацией с CAD системами и пакетами математических методов (MATLAB, MATRIX, EASY5).

Анализ динамических процессов различной физической природы методами численного интегрирования систем интегро-дифференциальных уравнений может быть выполнен при помощи программных комплексов MBTU и ПА-9 (МГТУ им. Н.Э. Баумана).

## 2. Прототипирование

В последние годы в процессе разработки технических изделий широко используются их физические прототипы. Быстрое прототипирование является важным этапом, как на этапе конструирования, так и в производственном цикле. Наличие прототипа позволяет наглядно оценить результаты геометрического моделирования, проанализировать параметры изделия, провести рекламную кампанию и исследовать рынок, использовать прототип на отдельных этапах изготовления изделия, например, при литье по выплавляемым моделям.

Изготовление прототипа можно выполнить различными способами, например, такими как: стерео литографии, LOM-технологии, при помощи термопринтера, ускоренного фрезерования и др. (рис.3).

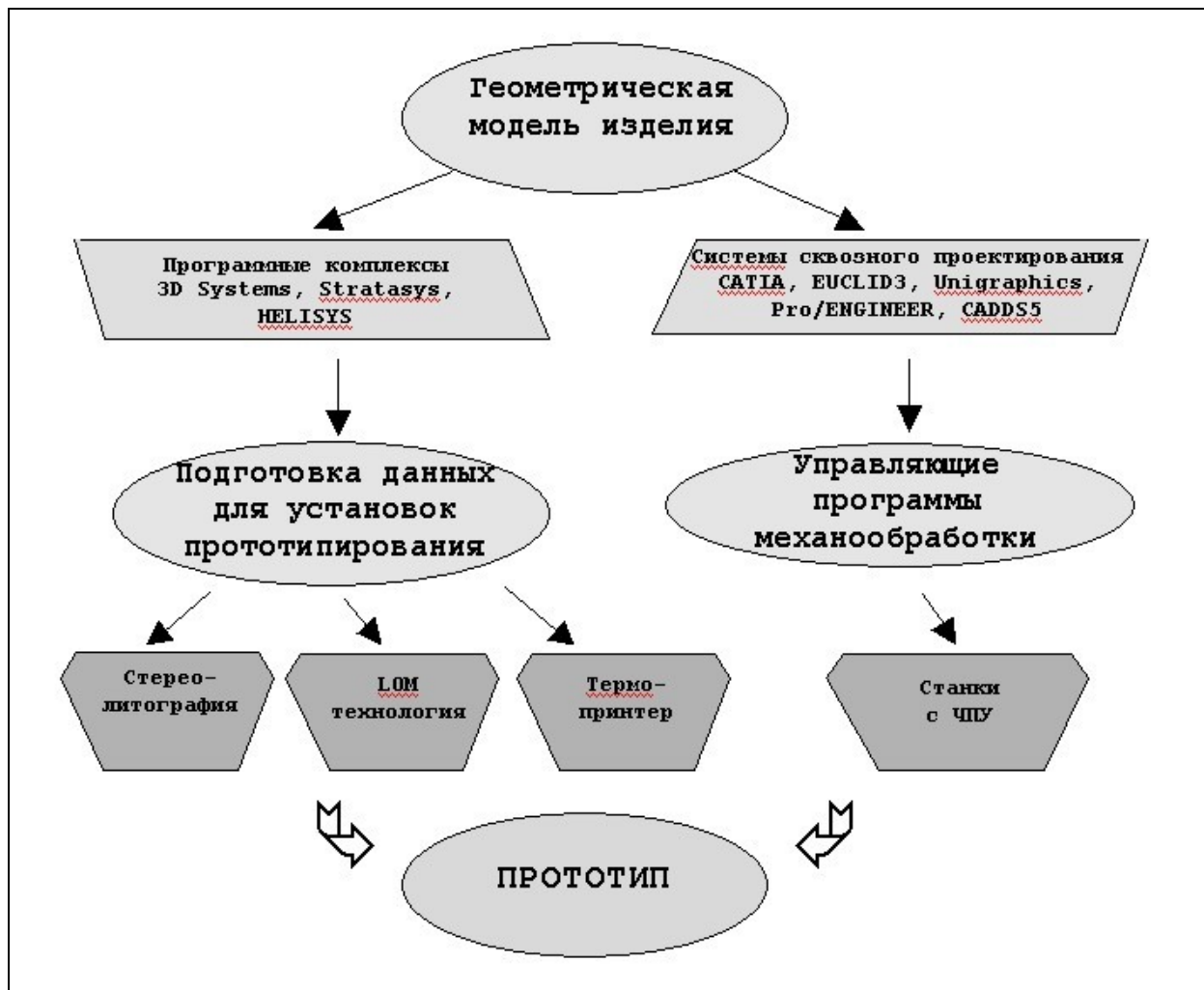


Рис. 3. Обобщенная схема процессов прототипирования.

Технологии всех методов прототипирования строятся на непосредственном использовании геометрической модели изделия. Например, для стерео литографии и технологии LOM при помощи специального интерфейса, который полностью интегрирован с системами сквозного проектирования, предварительно создается промежуточный файл в формате STL. Это позволяет получить доступ ко всем популярным платформам стерео литографических систем (например, фирм 3D Systems или Stratasys и др.), сохраняя при этом полную целостность данных. Данные STL-файла также могут быть использованы для механической обработки по технологии LOM фирмы HELISYS.

Для создания прототипов методом ускоренного фрезерования (гравирования) используются сравнительно недорогие мягкие материалы типа пластмасс, твердого пенопласта, дерева и т.п. Применение этих материалов позволяет существенно сократить время фрезерования при изготовлении прототипа. В качестве базовой модели также используется геометрическая модель изделия.

### 3. Технологическая подготовка производства

Этап технологической подготовки производства тесно связан с предыдущими этапами не только общими информационными, но часто и программными средами (рис.4).

Назначение этого этапа сводится к решению следующих основных задач:

- разработка технологий изготовления изделия, электродов, пресс-форм и штампов на основе их геометрических моделей, полученных на этапе проектирования;



- подготовка программ для станков с ЧПУ по спроектированным технологиям;
- контроль качества работы управляющих программ для станков с ЧПУ;

В производстве машиностроительных и части приборостроительных изделий используются технологии, в основе которых лежат различные физические процессы: механообработка, электроэрозионная обработка, литье металлов и пластмасс и др.

В автоматизированных системах сквозного проектирования и подготовки производства наиболее часто реализованы следующие виды механообработки: 2,5-й, 3-х и 5-ти координатное фрезерование, токарная обработка, сверление, нарезание резьбы и др. Имеется возможность моделировать движение инструмента и снятие материала во время черновой и чистовой обработки поверхности изделия. Например, в простейшем варианте 2-2.5 координатной обработки во многих программных комплексах реализованы следующие способы обработки поверхностей: контурная обработка, фрезерование призм и тел вращения, выборка «карманов» с возможностью движения «в одну сторону», зигзаг, спираль, а также нарезание резьбы и снятие фасок. В модулях 3- и 5- координатного фрезерования программных систем сквозного проектирования и ТПП реализованы практически все возможные способы обработки всех поверхностей изделий, например, такие как фрезерование поверхности с управлением углом наклона инструмента, шлифующее резание с возможностью обдувки и многие другие.

При выполнении различных видов механообработки используется общая база данных для поддержки связи между геометрической моделью обрабатываемой детали и управляющей программой для станка с ЧПУ, где проходы инструмента создаются по геометрии модели. Изменение геометрии отражаются в управляющей программе. Траектория движения инструмента создается интерактивно по поверхности модели изделия, обеспечивая технологов возможностью визуально наблюдать на мониторе имитацию процесса удаления стружки, контролировать зарезы и быстро вносить изменения в циклы обработки.

При помощи специальных программ автоматически вычисляется объем, который необходимо удалить из заготовки при обработке изделия.

Библиотека станков и режущих инструментов может быть расширена путем включения специфических для предприятия режущих инструментов, крепежных деталей и оснастки.

Электроэрозионная обработка приходит на смену традиционным металлорежущим технологиям и особенно широко применяется при изготовлении пресс-форм, вырубных и чеканочных штампов. В системах технологической подготовки производства могут быть реализованы как копировально-прошивочной, так и вырезной вид ЭЭО по 2-м - 5-ти координатам.

Можно выделить следующие основные этапы изготовления изделий с использованием метода ЭЭО:

- Автоматизированная разработка технологии изготовления электродов-инструментов на фрезерных станках с ЧПУ на основе геометрической модели изделия;
- Изготовление формообразующих элементов штампа на электроэрозионных станках с ЧПУ с использованием изготовленных электродов;
- Изготовление изделия методом штамповки.

Контроль данных для черновой, чистовой и окончательной обработки выполняется при помощи соответствующей программы.

Аналогичные этапы можно выделить при изготовлении изделий методами гальванопластики и литья по выплавляемым моделям.

Подготовка программ для всех видов оборудования с ЧПУ выполняется автоматически, когда выбран станок и указан тип процессора, установленный на данный станок (например, CNC или FANUC).

Обобщенная структурная схема этапа технологической подготовки производства и примеры программных сред, реализующих его основные задачи, приводится на рис.4.

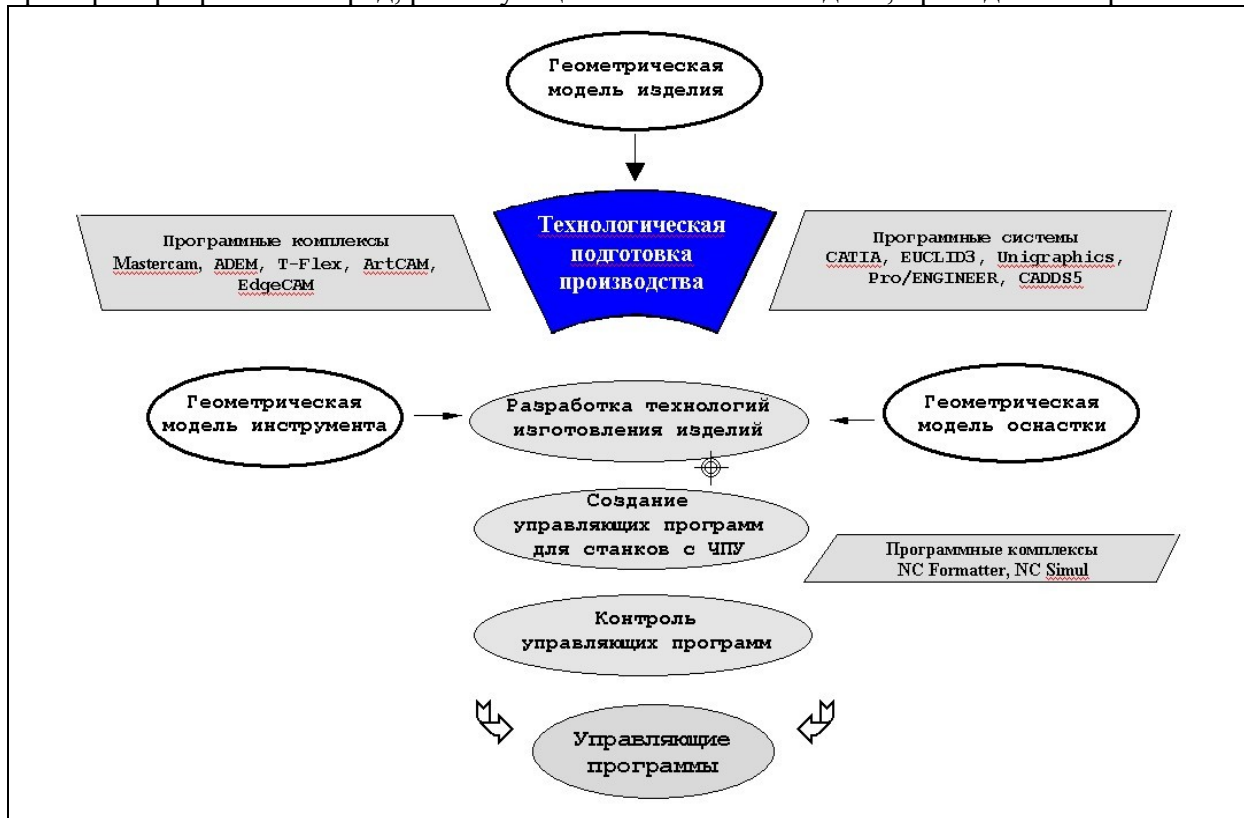


Рис. 4. Обобщенная схема этапа технологической подготовки производства.

Дополнительно остановимся на некоторых разработках российских фирм:

- - КОМПАС АВТОПРОЕКТ (Аскон) - проектирование технологических процессов механообработки, штамповки, сборки, термообработки и т.д.,
- T-FLEX ТехноПро (Топ Системы) - проектирование технологии механообработки, сборки, сварки, пайки, нанесения покрытий, штамповки,ковки термообработки,
- СИТЕП МО (Станкин СОФТ)-механообработка, СИТЕП ЛШ - листовая штамповка,
- TECHCARD (Интермех) - комплексная система автоматизации технологической подготовки производства,
- ТехноПро (Вектор) - универсальная система автоматизации технологического проектирования,
- SprutCAM СПРУТ-ТП (СПРУТ-Технология) система автоматизированного проектирования технологических процессов и др.

#### 4. Производство

В последнее годы в специальной литературе все чаще используется понятие «подвижное» производство или «виртуальное» предприятие. Под «виртуальным» предприятием понимается взаимодействие множества агентов, находящихся между собой в отношениях партнерства, кооперации, координации и т.п. Являясь элементами производственной структуры, они реализует совместный проект (или последовательность взаимосвязанных проектов), обеспечивая выполнение этапа производства изделия. Примером организации «подвижного» производства может быть объединение множества предприятий с целью создания международной космической станции. Эти же предприятия могут стать полноправными партнерами других «подвижных» производств. Выше отмечалось, что наибольшая производительность достигается в тех процессах разделения труда и на тех предприятиях, которые используют автоматизированные системы CALS технологий, принимают на себя ответственность за сохранность

получаемой информации, соблюдают авторские права на эту информацию и лицензионную чистоту программного обеспечения, способны оперативно переналаживать свое производство, гарантируют требуемое качество своей продукции и др. Соответствие современным требованиям организации производства и бизнеса отражается в международной сертификации предприятий.

### **Редактор представления данных**

Графический интерфейс операторских станций создается в редакторе представления данных. Его вид показан на следующем рисунке.

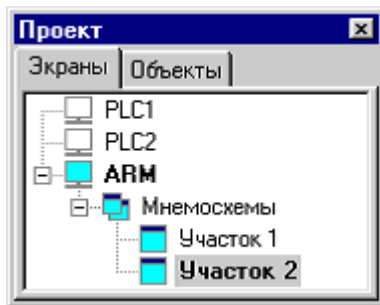


Редактор представления данных имеет следующие компоненты:

- Рабочая область;
- Навигатор проекта;
- Диалог настройки атрибутов графических элементов;
- Главное меню;
- Панели инструментов;
- Строка статуса.

### **Навигатор проекта**

Навигатор проекта используется для управления разработкой графического интерфейса операторских станций. Его вид представлен на следующем рисунке.



Навигатор проекта имеет два бланка. Первый из них **Экраны** предназначен для работы с графическими базами, а второй **Объекты** – для работы с библиотеками графических объектов.

При запуске редактора в навигаторе проекта открывается бланк **Экраны**. В нем после загрузки проекта выводится список узлов и содержание их графических баз: список групп и экранов, представленный в виде дерева. Нажатие ПК на любом из элементов данного дерева выводит на экран меню работы с этим элементом.

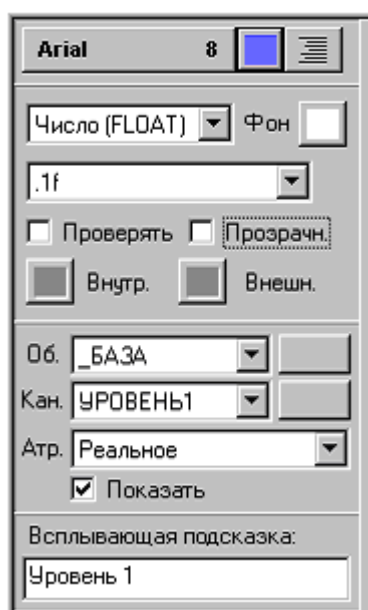
В бланк **Объекты** выводится список загруженных библиотек и графических объектов. Нажатие ПК области данного бланка вызывает на экран меню настроек библиотеки и графических объектов.

#### Диалог атрибутов графических элементов

Все формы отображения и элементы рисования имеют свои настройки. К ним относятся, например, параметры шрифта, имя переменной, с которой осуществляется связь, цвет и пр. Эти настройки графических элементов называются их атрибутами.

Атрибуты графических элементов редактируются в диалоге **Атрибуты**. Он выводится на экран при нажатии ЛК на любой пиктограмме одной из описанных выше инструментальных панелей. При этом диалог **Атрибуты** располагается в левой части экрана редактора под навигатором проекта.

Содержание этого диалога зависит от выбранного графического элемента. Пример диалога **Атрибуты** приведен на рисунке.



## Инструментальные панели

Редактор представления данных имеет следующие инструментальные панели:

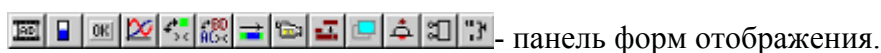
Системная панель;

Панель групповых операций (Сервис);

Панель статических элементов;

Панель форм отображения.

Следующий рисунок демонстрирует эти инструментальные панели.



В системную инструментальную панель вынесены команды управления режимами работы редактора представления данных, а также ряд команд из меню **Проект** и **Правка**.

В панель **Сервис** вынесены команды группового редактирования выделенных графических элементов и их тиражирования.

Панели элементов рисования и форм отображения используются для выбора размещаемых графических элементов. Почти всем пиктограммам в этих инструментальных панелях соответствуют группы графических элементов.

Для настройки пиктограммы на нужный элемент надо дважды нажать на ней ЛК. При этом на экране появляется меню, каждая пиктограмма которого обозначает графический элемент данной группы. После выбора элемента его пиктограмма появится в соответствующей ячейке панели. Далее при выборе этой ячейки будет осуществляться доступ к элементу, поставленному ей в соответствие.

## Режимы редактирования

Двойное нажатие ЛК в бланке навигатора проекта **Экраны** на имени экрана выводит его в рабочее поле редактора. Если разрешение экрана превосходит размеры рабочего поля, то в окне редактирования появятся линейки прокрутки.

Редактировать экран, не видя его полностью, не всегда удобно. Иногда требуется разместить крупный графический элемент (например, рамку вокруг экрана). При этом надо видеть весь экран.

Для этого предусмотрен полноэкранный режим редактирования. Вход в этот режим осуществляется либо по нажатию сочетания клавиш **Ctrl-F**, либо командой **Полный экран** из меню **Вид**. При этом с экрана исчезает заголовок окна редактора и его меню.

Остальные инструменты остаются на экране. Они принимают вид диалогов и могут произвольно перемещаться по экрану. Таким образом, все поле экрана становится доступным для редактирования.

Возврат из этого режима к нормальному виду окна редактора представления данных осуществляется по повторному нажатию **Ctrl-F**.

### ***Элементы графического интерфейса***

При разработке графического интерфейса в ТРЕЙС МОУД используются следующие термины:

- элемент рисования;
- форма отображения;
- экран;
- графический объект;
- графическая база.

#### **Элемент рисования**

Элементами рисования называются элементарные графические фрагменты, из которых составляется статическая часть графического оформления экрана.

Статической называется неизменяемая часть оформления экрана. Она не зависит от значений контролируемых параметров и состояния автоматизируемого технологического процесса (изображения аппаратов, рисунки приборов и пр.).

Статический рисунок составляется путем размещением в поле экрана элементов рисования. Для их выбора используется соответствующая инструментальная панель редактора.



После настройки атрибутов элемента рисования его можно разместить на экране. Для большинства элементов последовательность операций по их размещению выглядит следующим образом:

- поместить курсор мыши в рабочее поле и нажатием ЛК выбрать точку привязки;
- перемещая курсор мыши, добиться требуемого размера графического элемента;
- нажать ЛК повторно.

Размещение некоторых элементов рисования осуществляется в несколько этапов. Подробное описание всех элементов рисования, их атрибутов и особенностей размещения будет приведено в разделе **Визуализация технологического процесса**.

#### **Форма отображения**

Формы отображения – это графические динамические элементы, позволяющие отобразить значения контролируемых технологических параметров и реализовать супервизорное управление ими.

Формы отображения выбираются с помощью соответствующей панели инструментов редактора представления данных.



Настройка формы отображения заключается в задании ее атрибутов и связывании ее с отображаемой переменной.

После настройки формы отображения ее можно размещать на экране. Для большинства форм отображения последовательность операций по их размещению выглядит следующим образом:

- поместить курсор мыши в рабочее поле и нажатием ЛК выбрать точку привязки;
- перемещая курсор мыши, добиться требуемого размера графического элемента;
- нажать ЛК повторно.

Размещение некоторых форм отображения отличается от описанного выше. Подробное описание всех форм отображения, их атрибутов и особенностей размещения приведено в разделе **Визуализация технологического процесса**.

## Экран

### [Атрибуты экрана](#)

### [Группы экранов](#)

Графический интерфейс операторских станций разрабатывается в виде отдельных экранов.

Экран - это графическое пространство фиксированного размера, на котором размещается статический рисунок и формы отображения. Экран - это минимально адресуемая графическая информация, выводимая на монитор.

Одновременно на монитор может выводиться только один экран. Другие экраны можно просматривать на текущем экране с помощью **окон** - специальных форм отображения.

### Атрибуты экрана

Каждый экран имеет свое имя и набор атрибутов. К таким атрибутам относятся:

- Размер;
- Цвет фона;
- Обои;



Права доступа;

Спецификация окна просмотра отчета тревог.

Первые три атрибута по умолчанию устанавливаются по общим настройкам графической базы. Права доступа к экрану устанавливаются полными.

## Группы экранов

Все экраны каждого узла разбиты на группы. Каждая группа имеет свое название. Группировка экранов не используется для их адресации при работе в реальном времени. Она удобна для ориентации в больших графических базах, включающих в себя сотни экранов.

## Графический объект

При создании графического интерфейса для удобства тиражирования созданных элементов, а также для решения ряда других задач можно использовать графические объекты.

Графическим объектом называется совокупность форм отображения и элементов рисования, которая оформлена как единый графический элемент.

Существует два типа графических объектов: **Объект** и **Блок**. Первый из них может ссылаться на 256 каналов, а последний - только на один.

Графические объекты собираются в библиотеки. При создании объекта надо указать библиотеку, в которой он будет сохранен. Для этого ее надо либо загрузить в редактор, либо создать.

Для создания и редактирования объектов используются такие же окна, как и при работе с экранами. Разработка объектов также идентична процессу разработки экрана. Отличие заключается в настройке форм отображения на каналы. В объекте ФО связываются с его внутренними каналами. Эти каналы при размещении объекта на экране настраиваются на реальные каналы редактируемого узла.

ТРЕЙС МОУД позволяет осуществлять ряд операций с графическими объектами. Среди таких операций: копирование, сохранение и вставка в другие проекты или графические базы того же проекта, вывод в отдельные окна на других экранах и пр.

## Графическая база

Графическая база - это совокупность всех экранов, графических объектов, элементов рисования и форм отображения, используемых для отдельного узла проекта.

Графические базы создаются для всех операторских станций проекта. Сохранение графических баз осуществляется в файлах с расширением **dbg** в директории редактируемого проекта.

Имя файла графической базы для узла задается в редакторе базы каналов. Для этого используется бланк **Основные** диалога **Параметры узла**. Вход в этот диалог



осуществляется нажатием ПК на изображении узла в окне редактирования структуры проекта.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 7. [Языки разработки алгоритмов Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»:**

**- описание функциональных блоков (управление, ввод/вывод, переходы, обобщение, LD-функции, регулирование, пересылки).**

## ЛЕКЦИЯ 8.

### ЕДИНОЕ ИНФОРМАЦИОННОЕ ПРОСТРАНСТВО ПОДДЕРЖКИ ЖИЗНЕННОГО ЦИКЛА ИЗДЕЛИЯ

1. Стандарты
2. Управление данными в автоматизированных системах
3. Архивирование в Trace Mode 5
  - общие положения;
  - СПАД – локальный архив.

#### 1. Стандарты

Автоматизированные системы (АС), поддерживающие тот или иной этап жизненного цикла изделия, как правило, имеют иерархическую многоуровневую структуру. На каждом уровне функционирования АС используются определенные модели и способы обмена данными между компонентами АС. Причем эти модели и способы обмена должны быть согласованными во всех взаимодействующих компонентах. Очевидно, что если взаимодействие касается только внутренних компонентов одной конкретной АС вопросы согласования целиком находятся в компетенции разработчиков этой АС.

Иначе обстоит дело при взаимодействии на верхних иерархических уровнях, где обмен информацией происходит между разными АС, зачастую не имеющих единой организационной структуры, т.е. в рамках «подвижных» производств или «виртуальных» предприятий.

Создание единой информационной среды для функционирования разных САЕ, САМ, САЕ, PDM, ERP, MES, CSM и т.п. систем базируется на разработке и использовании совокупности международных и национальных стандартов. При этом унификация спецификаций должна касаться как онтологии приложений, так и форм представления моделей.

Онтологией принято называть систему понятий (сущностей и их атрибутов) в конкретной предметной области и отношений между этими понятиями.

При разработке онтологий рекомендуется начинать с определения набора объектов предметной области, достаточно полного для моделирования разных изделий данного приложения. Основные группы выбранных объектов называют интегрированными ресурсами, которые подразделяются на общие (используемые во многих приложениях) и прикладные. Обеспечение полноты и непротиворечивости модели достигается путем использования методик функционального и информационного моделирования, закрепленных в международных стандартах IDEF0 и IDEF1X.

После достижения соглашений о содержимом моделей приложений необходимо обеспечить их представление на едином языке, понятном всем исполнителям и участникам этапов жизненного цикла изделий. Известно большое число форматов данных, предлагавшихся в процессе развития автоматизированных систем для целей представления и обмена данными. Наиболее удачные из них принято считать компонентами CALS-технологий.

Например, в стандарте министерства обороны США MIL STD 1840C Automated Interchange of Technical Information (1997 г.), посвященном технической поддержке обмена большими объемами данных между гетерогенными компьютерными системами, приведен перечень ряда стандартов, относящихся к CALS, и среди них, в частности, названы следующие документы:

- ANSI X3.4-1986 - American National Standard Code for Information Interchange (известный формат ASCII);
- FIPS PUB 46-2 - Data Encryption Standard (DES) - шифрование данных;
- FIPS PUB 186 - Digital Signature St - электронная подпись;

- EIA standard EDIF - Electronic Design Interchange Format - формат для представления данных о тестах и соединениях печатных плат и кристаллов БИС в производстве;
- IEEE 1076 - VHDL (Very high-speed integrated circuits Hardware Design Language) - язык моделирования дискретных электронных устройств;
- MIL PRF 28000 - IGES Application subset and IGES Application protocols - формат для представления данных об изделиях;
- MIL PRF 28001 - markup requirements and generic style specification for exchange of text and its presentation - требования разметки и стили спецификаций для представления и обмена данными;
- MIL PRF 28002 - raster graphics representation in binary format, requirements for - представление растровой графики в двоичном формате;
- MIL PRF 28003 - digital representation for communication of illustration data: computer graphic metafile (CGM) application profile - метафайл графических данных;
- MIL PRF 28000B (1999) - Digital Representation for Communication of Product Data: IGES Application Subsets and IGES Application Protocols. В этой версии стандарта установлены семь классов моделей IGES:
  - Класс 1: технические иллюстративные данные;
  - Класс 2: чертежные данные;
  - Класс 3: электрические/электронные приложения;
  - Класс 4: геометрия для NC;
  - Класс 5: прикладной протокол для трехмерных систем трубопроводов;
  - Класс 6: развитие класса 3 с упором на производство и сборку;
  - Класс 7: 3D геометрия.

Однако эти стандарты или касались частных задач или имели заметную прикладную направленность, в них не был описан единый для разных приложений язык представления и обмена данными. Поэтому основным стандартом, составляющим сущность CALS-технологий, стал стандарт ISO 10303 с названием STEP (Standard for Exchange of Product model Data), состоящий из нескольких десятков томов.

В стандарте STEP установлен единый язык Express для представления моделей на всех этапах жизненного цикла изделий разных приложений. Этому языку вместе с его графической версией Express-G посвящен один из первых томов стандарта ISO 10303. В других томах стандарта с использованием Express описываются способы обмена данными, интегрированные ресурсы, способы проверки корректности Express-моделей. Представлены также онтологии ряда приложений, выраженные средствами языка Express и называемые в STEP прикладными протоколами (AP - Application Protocols). Разработан ряд прикладных протоколов для машиностроения, судостроения, инженерных служб в строительстве, концептуального проектирования предприятий нефтегазовой и химической промышленности, проектирования и монтажа электротехнических изделий, компоновки и проектирования межсоединений в электронной аппаратуре, обработки деталей на станках с ЧПУ и др.

К числу прикладных протоколов, наиболее часто реализуемых в современных CAD/CAM системах в машиностроении, относится протокол AP203. В нем унифицированы геометрические модели, атрибуты и спецификации сборок, параметры управления версиями и внесением изменений в документацию в процессе проектирования и др. В стандарте дано описание ряда сущностей, относящихся к конфигурации изделия, например, таких как вносимое в проект изменение с соответствующими атрибутами. Для геометрического 3D моделирования в протоколе установлено шесть классов моделей. Класс 1 предназначен для задания состава изделий без описания геометрических форм. Класс 2 включает каркасные модели с явным описанием границ, например, в виде координат точек и определяемых с их помощью линий. В классе 3 каркасные модели дополнены топологической информацией, т.е. данными о том, как поверхности, линии или точки связаны друг с другом. Класс 4 служит для описания поверхностей

произвольной формы. Классы 5 и 6 включают твердотельные модели, так называемые BREP (Boundary representation). К классу 5 относятся тела, границы которых аппроксимированы полигональными (фасеточными) поверхностями, состоящими из плоских участков. В классе 6 поверхности, ограничивающие тела, могут быть как элементарными (плоскими, квадратичными, тороидальными), так и представленными моделями в форме Безье, B-сплайнов и др.

Если язык Express используется для представления информационных и функциональных моделей приложений, то язык SGML (стандарт ISO 8879 - Information Processing - Text and Office Systems - Standard Generalized Markup Language) предназначен для унификации представления текстовой информации в АС - разнообразных проектных и эксплуатационных документов. Стандарт SGML устанавливает такие множества символов и правил для представления информации, которые позволяют правильно распознавать и идентифицировать информацию. Эти множества символов описываются в отдельной части документа, называемой декларацией DTD (Document Type Definition), которую помещают в SGML-документ вместе с основным текстом. В DTD указывают соответствие символов и их кодов, максимальные длины используемых идентификаторов, способ представления ограничителей для тегов, другие возможные соглашения, синтаксис DTD, а также тип и версию документа. Следовательно, SGML можно назвать метаязыком описания семейства конкретных языков разметки. В настоящее время перспективным языком разметки в CALS-технологиях считают язык XML, являющийся более удобным в использовании подмножеством SGML.

К числу основных CALS-стандартов относятся также стандарты Parts Library (ISO 13584) и Mandate (ISO 15531).

Стандарт ISO 13584 содержат обзор и основные принципы представления данных о стандартных компонентах промышленных изделий. В этих стандартах представлены в виде библиотек с использованием языка Express данные о таких типовых компонентах изделий, как болты, подшипники, электронные компоненты и т.п., с целью использования этих данных в системах автоматизированного проектирования.

Стандарт Mandate посвящен представлению данных, относящихся к функционированию предприятий, управлению территориально распределенными производственными системами, обмену данными о производстве с внешней для предприятия средой.

## **2. Управление данными в автоматизированных системах**

Сложность технических изделий, наличие множества их модификаций, заимствование, стандартизация, унификация, требуют поддержки многоуровневых, многовариантных моделей. Нарастает мощность информационных систем предприятий, растет количество рабочих мест дизайнеров, технологов и других специалистов, которые работают в различных программных средах. Этапы жизненного цикла изделия поддерживаются множеством предприятий, и организация эффективного обмена и управления информацией остается актуальной. Проблемы управления моделями и процессами отдельных компаний и «виртуальных» предприятий являются сферой интересов CALS-технологий. Среди них можно выделить наиболее важные:

- ведение распределенных архивов конструкторской, технологической, коммерческой информации (геометрические модели, конечно-элементная сетки, результаты инженерного анализа, конструкторская документация, технологические процессы, программы для станков с ЧПУ, экономические расчеты и т.д) ( рис. 5);

- организация быстрого поиска, просмотра и аннотирования документов и моделей различных форматов (текстовой информации, графической, вербальной, анимация и пр.) без загрузки приложений, в которых эта информация создавалась;

- управление разработками сложных изделий (управления структурой изделия, согласование процедур и этапов работ, управление версиями и документами, история создания и сопровождения);
- организация обмена и представления данных различных CAD/CAM/CAE и MRP-систем;
- авторизация пользователей и рабочих групп, назначение прав доступа и защита информации;
- установление последовательности прохождения документов, контроль выполнения, электронная подпись;
- организации информации на физических носителях различного типа, удаленный доступ к информации.



Рис. 5. Единое

*информационное пространство поддержки жизненного цикла изделия.*

В современных CALS технологиях эти и другие функции берут на себя программные среды, получившие наименование PDM/ EDM (Product Data Management) / (Enterprise Data Management). PDM предназначены для управления информацией об изделии на всем жизненном цикле (разработка, производство и эксплуатация). EDM принимают на себя функции управления информацией о процессах.

Исторически сложились три группы разработчиков этих систем:

- ◇ создатели программных системы сквозного проектирования и производства, разработки которых дополняют ранее созданные среды PDM-системами. Например, CATIA Product Manager (Dassault Systemes), Optegra (PTC), Особенности таких систем состоят в том, что они ориентированы, в первую очередь, на обслуживание своих CAD/CAM приложений;
- ◇ вторую группу составляют разработчики систем управления производством, т.е. MRP/ERP-систем, которые стремятся объединить представление об изделии в средах разработки и изготовления;
- ◇ независимые разработчики, деятельность которых направлена на создание единой информационной системы путем объединения большинства CAD/CAM и MRP/ERP-систем.

Приведем примеры некоторых систем PDM различных разработчиков:

- Unigraphics Information Manager (IMAN) фирмы Electronic Data Systems (EDS) - одна из наиболее универсальных систем управления информацией;
- BAAN V PDM - совместная разработка фирм Baan и BAIN (B.A. Intelligence Networks) - полностью интегрированная PDM/ERP - система;
- EUCLID Design Manager фирмы EADS Matra Datavision - первая PDM система этой фирмы, разработанная в качестве составляющей программного комплекса EUCLID Quantum и EUCLID3;
- CENTRA 2000 фирмы Auto-trol - объединяет систему управления конфигурацией изделий на всем жизненном цикле и систему управления документами и проектами;
- Metaphase - результат совместной разработки фирм SDRC и CDS. Система отличается высокой степенью интеграции с CAD/CAM I-DEAS, AutoCAD, Pro/Engineer и др.;
- OPTEGRA - система фирмы Computervision объединяет три модуля: Total Data Management (TDM) - управление данными предприятия, Workflow Management (WM) - управление процессами и Configuration Management (CM) - управление конфигурацией изделия;
- ProductManager - разработка IBM, ориентированная на использование серверов на RISC-платформах IBM, SUN, HP в СУБД Oracle или DB2/6000 и др.

Во многих системах PDM в той или иной мере реализованы схожие функции. На примере одной из наиболее распространенных систем DMS/PIMS фирмы Sherpa Corp. рассмотрим возможности таких систем. Корпорация Sherpa специализируется на разработке PDM систем и ее пакет DMS/PIMS инсталлирован на более чем 50000 рабочих мест в таких фирмах, как: AT&T, McDonnell, Douglas, General Motors, MATRA, Fiat Avio, Siemens, Philips, Saab и многих других. К особенностям этого пакета можно отнести:

- эффективная маршрутизация документов,
- авторизация пользователей и управление доступом к данным,
- наличие средств удаленного редактирования данных, электронная подпись,
- формирование, ведение и обработка спецификаций,
- автоматизация управления изменением данных об изделиях,
- организация совместной работы над проектом в составе рабочих групп,
- интеграция различных программных систем и комплексов:
  - ◆ системы сквозного проектирования и подготовки производства машиностроительных изделий - CATIA, EUCLID3, UNIGRAPHICS, Pro/ENGINEER, CADD5, I-DEAS, AutoCAD;
  - ◆ программные среды схемотехнического проектирования фирм Mentor Graphics, Cadence;
  - ◆ ERP (АСУ) - системы - ASK MANMAN-X, SAP R/3, Oracle Manufacturing;
  - ◆ Офисные приложения - Excel, Word (Microsoft).

В настоящее время фирма Sherpa ориентирует свои разработки на управление компаниями, международными корпорациями и промышленными группами.

Сравнительная оценка функциональности многих PDM систем приводится в [4].

В последние годы активно ведутся разработки программных систем, способных обеспечить совместную работу над проектом в режиме реального времени с использованием сети Internet. Примерами таких систем являются DesignKnet (Design Knowledge Network) фирмы Unigraphics Solutions, Windchill фирмы PTC и др.

К основным требованиям, предъявляемым к таким системам, можно отнести:

- функционирование в стандартных Web технологиях,
- способность поддерживать взаимодействие с различными приложениями, использующих различные базы данных,
- визуализация данных из различных CAD/CAM систем должна поддерживаться даже на персональных компьютерах,

- адаптация и развитие системы не должны требовать разработки дополнительных программ и усилий специалистов фирм-разработчиков и др.

Если с внедрением PDM систем ставится вопрос о том, какая технология наилучшим образом подходит для управления процессом разработки изделия, то с развитием последних систем пользователи могут оценивать поставщиков и технологии в более широком диапазоне выбора решений, наиболее подходящих для получения максимальной отдачи при использовании интеллектуального потенциала на всем глобальном пространстве.

#### **Список литературы.**

1. А.Солдаткин, Ю.Голенков, В.Кононов, И.Караулов. Программа MSC.SuperForge как один из элементов системы виртуального производства и управления качеством изделий. САПР и графика №7. 2000.

2. А.Мазурин. Моделирование холодной и горячей объемной штамповки в Qform. САПР и графика № 8. 2000.

3. В.Бойков. Моделирование динамики механических систем в программном комплексе Euler. САПР и графика №1. 1998.

4. С.Локшин, С.Ротков. Системы PDM наWeb-серверах Internet. САПР и графика №1. 1998.

### 3. Архивирование

#### **Общие положения**

В ТРЕЙС МОУД поддерживаются три типа архивов. Кроме того, используя ODBC, можно сохранять информацию в любые базы данных, поддерживающие этот протокол.

Сохранение информации в архивах настраивается при конфигурировании системы. Однако многие настройки могут меняться в реальном времени с помощью специальных каналов.

#### **Организация сохранения данных**

[СПАД - локальный архив](#)

[Отчет тревог](#)

[Глобальный регистратор](#)

ТРЕЙС МОУД поддерживает три архива:

СПАД (локальный архив);

Отчет тревог;

Глобальный регистратор.

Разница между архивами заключается в алгоритме сохранения данных и формате файлов.

Архив может быть размещен на удаленном компьютере через "сетевой диск". Использование имени удаленного компьютера для задания пути к архиву не предусмотрено, в этом случае выдается сообщение об ошибке и архив не открывается.

Кроме того, некоторые мониторы ТРЕЙС МОУД поддерживают локальный архив в ОЗУ и энергонезависимой памяти.

### **СПАД - локальный архив**

В этот архив значения каналов записываются в бинарном формате. Условием записи является изменение значения канала. СПАД имеет фиксированную длину. При этом глубина архивирования определяется заданным размером и интенсивностью потока данных.

### **Отчет тревог**

Отчет тревог ведется в ASCII-формате. Он предназначен для фиксации событий. Теоретически его размер не ограничен. Данные заносятся в отчет тревог в виде строк. Каждая строка содержит время и дату ее формирования, а также ряд дополнительных полей. Эти поля могут содержать разную информацию в зависимости от типа фиксируемого события. Такой информацией может быть, например, имя канала и его значение, сообщение из словаря, время квитирования и пр.

### **Глобальный регистратор**

Глобальный регистратор – это архив, который ведет специализированный монитор. Значения архивируемых в регистраторе каналов посылаются ему по сети при их изменении.

Монитор глобального регистратора имеет фиксированный групповой номер в сети - 200. Архивируемые значения каналов посылаются на этот сетевой номер. Таким образом, в рамках проекта может существовать только один регистратор. Однако количество его дублей не ограничено. Все присутствующие в сети мониторы регистраторов будут одновременно принимать данные, посылаемые для сохранения. Естественно, каждый из них будет вести свой файл архива, но все эти файлы будут идентичны.

Максимальная точность фиксации времени в архивах ТРЕЙС МОУД равна 1 миллисекунде.

### ***Настройка каналов для архивирования***

Чтобы значения канала записывались в любой из архивов ТРЕЙС МОУД, необходимо установить ему соответствующие флаги.

Флаги архивирования устанавливаются в бланке **Основные** диалога **Реквизиты**. Для входа в него надо дважды нажать ЛК на имени канала в **Каналы объекта** любого объекта базы каналов. На рисунке показаны поля для установки флагов архивирования.





### **Настройка архивов**

Чтобы значения каналов архивировались, недостаточно только установить для них соответствующие флаги. Надо еще указать параметры архивов, в которые будут сохраняться значения каналов.

Настройка архивов осуществляется в бланке **Архивация** диалога **Параметры узла**. Вход в этот диалог осуществляется из окна редактирования структуры проекта по нажатию ПК на графическом идентификаторе узла.



### **СПАД - локальный архив**

Этот архив предусмотрен для сохранения на диск и последующего анализа значений каналов текущего узла. В нем фиксируются изменения реальных значений и не вычисляемых атрибутов каналов.

### **Организация записи данных в СПАД**

При настройке СПАД-архива задается имя файла, в котором будут сохраняться данные, и его размер в мегабайтах.

Монитор реального времени получает доступ к данным из СПАД с помощью архивных трендов. Мониторы **SUPERVISOR** и серверы документирования также получают данные из этого архива. Кроме того, имеется ODBC-драйвер для доступа к архивным данным из других приложений.

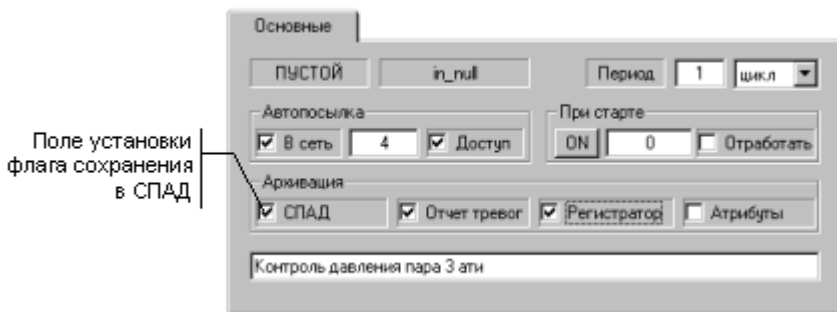
## Условия сохранения

Значения канала записываются в СПАД по изменению. При этом в архив добавляется одна запись, фиксирующая новое значения и время. Точность фиксации времени составляет 1 мс.

Время записи равно базовому времени цикла пересчета базы. Это означает, что при многократных изменениях какого-либо архивируемого атрибута в пределах одного цикла пересчета базы в архив попадет значение последнего изменения.

В СПАД заносятся значения каналов, для которых установлены соответствующие флаги. Флаг **СПАД** устанавливается в бланке **Основные** диалога **Реквизиты**. Для сохранения в любые архивы не только реального значения канала, а так же изменений его остальных не вычисляемых атрибутов, следует в том же бланке установить флаг **Атрибуты**.

Вид этого бланка **Основные** диалога **Реквизиты** показан на следующем рисунке.



## Глубина сохранения данных

Глубина хранения данных определяется размером архива и интенсивностью потока данных. Поскольку размер архива ограничен, то увеличение времени хранения достигается сокращением интенсивности потока данных. Для этого следует вводить апертуры по каналам, что позволяет не фиксировать малые изменения, а для инерционных параметров можно увеличивать период опроса.

Количество записей в архиве определяется его размером, длиной записи и размером заголовка. Величина одной записи равна 16 байтам, а размер заголовка приблизительно 1 Мбайт.

## Механизм сохранения данных в СПАД

Данные в СПАД обновляются циклически. Перед добавлением новой записи контролируется ее положение в файле. Если места для записи больше нет, то она записывается в начало архива. Далее все новые записи записываются поверх самых дальних по времени.

Сохранение данных в СПАД реализовано в виде потока, работающего параллельно с пересчетом базы каналов, но с более низким приоритетом. МРВ формирует внутреннюю очередь сообщений для записи в СПАД. Поток архивирования берет данные из нее и записывает их в архив. Если размер очереди превышен, то самые дальние по времени сообщения теряются. По умолчанию максимальный размер очереди принимается равным 64000 сообщений.

Контроль состояния очереди сообщений в СПАД и управление ей осуществляется с помощью канала подтипа **ДИАГНОСТИКА** с дополнением **очередь СПАД** (см. ниже).

### **Инициализация СПАД**

МРВ, сохраняющий данные в СПАД, инициализирует его при первом запуске. Он проверяет наличие свободного места на диске. Если его недостаточно для открытия архива, то МРВ завершает свою работу с кодом ошибки 36. Если места на диске хватает, то файл архива создается. В нем прописывается заголовок, в котором формируются структуры для индексации данных в архиве. Размер заголовка локального архива равен приблизительно 1 Мбайт.

Если указанная длина архива меньше, чем размер заголовка, и на диске есть свободное место, то файл архива создается. Его размер будет равен 1.4 Мбайта. Это позволяет хранить 22770 записей. Размер одной записи в СПАД равен 16 байтам.

Если при запуске МРВ уже существует файл архива с тем же именем, то проверяется идентичность его структуры требуемой. При этом сравниваются: установленный размер и имя узла. Если существующий архив не соответствует требуемым параметрам, то МРВ завершает свою работу с кодом ошибки 36.

### **Настройка СПАД**

Параметры СПАД настраиваются в редакторе базы каналов. Для этого, находясь в окне редактирования структуры проекта, надо нажать ПК на графическом идентификаторе настраиваемого узла. В результате на экране появится диалог **Параметры узла**. Настройка параметров архивирования осуществляется в бланке **Архивация**. Вид этого бланка показан на следующем рисунке.



Для локального архива СПАД здесь указывается имя файла архива, путь к этому файлу и его размер в мегабайтах.

**Внимание!** Если задано только имя файла архива, а путь к нему не указан, то запуск проекта по сети (с удаленного диска) осуществить не удастся. В этом случае будет выдана ошибка "-36".

### **Контроль состояния и управление СПАД**

Для контроля и управления архивированием данных в СПАД предусмотрены следующие каналы: подтип **ДИАГНОСТИКА** с дополнениями [СПАД](#) , [Потеря СПАД](#) , [Очередь СПАД](#) ; подтип **СИСТЕМНЫЙ** с дополнениями [Архивация](#) , [СПАД копировать](#) ,

### Управление сохранением данных

Канал **СИСТЕМНЫЙ** с дополнением [Архивация](#) управляет сохранением во все архивы. Значение его 0-го бита управляет разрешением записи в СПАД, а 8-го разрешением открытия файла архива:

**0** – разрешить;

**1** – запретить.

Запрет открытия файла используется при записи архива на сменный носитель во время его замены. При этом файл закрывается, а новые данные, накапливаются в буфере. После замены носителя значение 8-го бита следует снова установить равным 0. В результате на новом носителе создается файл архива. В него сохраняются данные из буфера и процесс архивирования продолжится.

Принудительное сохранение данных в СПАД реализуется с помощью канала типа **OUTPUT** подтипа **Диагностика** с дополнением [Потеря СПАД](#) .

### Экспорт данных из СПАД

#### [Экспорт значений одного канала](#)

#### [Экспорт всех архивируемых каналов](#)

MPV может экспортировать данные из локального архива в файлы текстового формата. Существует возможность экспортировать архивные значения одного канала или всей базы целиком.

### Экспорт значений одного канала

Для управления экспортом используется канал типа **OUTPUT** подтипа **КАНАЛ** с дополнением [SetGet СПАД](#) . Он имеет настройки для выбора канала и его атрибута и настройку, задающую диапазон выборки. Его значение задает смещение базового времени в секундах относительно начала текущих суток. Диапазон выборки отсчитывается назад от полученного базового времени. Положительное значение канала задает смещение назад, а отрицательное - вперед.

**Внимание!** Из возможных временных диапазонов выборки архивных данных в канале [SetGet СПАД](#) не обрабатываются **за год и до выключения**.

Экпортируемые данные сохраняются в текстовый файл, имя которого образуется из имени указанного канала. При каждой операции экспорта новые данные дописываются в конец данного файла.

### Экспорт всех архивируемых каналов

Экспорт всех архивируемых каналов осуществляется в текстовый файл с именем **data.txt**. Он располагается в директории проекта. При каждой операции экспорта новые данные дописываются в конец файла. Данные в него заносятся в следующем формате:

<имя канала 1>

<дата время> <значение>

...

<дата время> <значение>

...

<имя канала n>

<дата время> <значение>

...

<дата время> <значение>

Для управления экспортом данных из СПАД используется канал типа **OUTPUT** подтипа **СИСТЕМНЫЙ** с дополнением [Данные из СПАД](#). Его значение следующим образом определяет временной диапазон выборки и вид представления экспортируемых каналов:

- 1 – за предыдущие сутки по каналам **F**;
- 2 – за предыдущие сутки по каналам **H**;
- 3 – за предыдущий час по каналам **F**;
- 4 – за предыдущий час по каналам **H**;
- 5 – за текущий час до текущей минуты по каналам **F**;
- 6 – за текущий час до текущей минуты по каналам **H**.
- 7 – за последние 24 часа по каналам **F**;
- 8 – за последние 24 часа по каналам **H**;
- 9 – за текущие сутки до текущего часа по каналам **F**;
- 10 – за текущие сутки до текущего часа по каналам **H**.

Данный канал типа **INPUT** контролирует чтение данных из **СПАД**. Его значение соответствует следующим состояниям:

- 0 – никаких действий не выполняется;
- 1 – запуск операции;

- 2 – выполнение операции;
- 4 – ожидание;
- 5 – завершение операции;
- 6 – ошибка выполнения операции;
- 7 – нормальное завершение операции.

### **Копирование СПАД**

Для управления копированием СПАД используется канал подтипа **СИСТЕМНЫЙ** с дополнением **СПАД копировать**. Посылаемое в него значение определяет путь к копии:

- 1 – в директорию проекта;
- 2 – в корневую директорию диска, с где записан проект;
- 3 – в корневую директорию диска **A**;
- 65-95 – в корневые директории дисков:
- 65 – **A**;
- 66 – **B**;
- ...

Имя файла копии архива образуется из 8-разрядного шестнадцатеричного числа, кодирующего дату и время (число секунд с 00:00:00 01/01/1970).

**Внимание!** Данные, записанные в архив во время его копирования, в копии отсутствуют.

### **Текущее состояние операций со СПАД**

Для контроля сохранения данных в СПАД и чтения из него предназначен канал типа **INPUT** подтипа **ДИАГНОСТИКА** с дополнением **СПАД**. Его значение соответствует следующим состояниям:

- 0** – ошибок нет;
- 1** – не найден файл архива;
- 2** – ошибка позиционирования головок в устройстве;
- 3** – ошибка записи;
- 4** – ошибка чтения;
- 5** – недостаточно памяти для выполнения операции;

6 – сбой при работе с очередью сообщений;

11 – ошибка выполнения системных функций.

Если этот канал имеет тип **OUTPUT**, то любая его отработка обнуляет признак текущего состояния операций с локальным архивом.

### Очередь сообщений в СПАД

При сохранении данных в СПАД формируется очередь записей. По умолчанию ее размер равен 128 блока по 1024 записи. Это требует 2М оперативной памяти. Изменить эту величину можно каналом подтипа **ДИАГНОСТИКА** с дополнением [Очередь СПАД](#). Его значение задает размер буфера в блоках по 1024 записи. Если установить тип **INPUT**, то такой канал контролирует величину очереди.

Поток архивирования берет данные из очереди и записывает их в СПАД. Если интенсивность потока данных превышает скорость записи их на диск, очередь начинает расти. При достижении максимального размера очереди новые сообщения записываются поверх самых старых.

Число потерянных сообщений контролируется каналом подтипа **ДИАГНОСТИКА** с дополнением [Потеря СПАД](#). Если он имеет тип **OUTPUT**, то изменение его значения обнуляет счетчик потерь.

### Пример

Решение задачи оперативного управления уровневый архивом.

Использование канала подтипа **ДИАГНОСТИКА** с дополнением к подтипу [СПАД](#).

Использование канала подтипа **СИСТЕМНЫЙ** с дополнением к подтипу [Архивация](#).

Использование языка функциональных блоков.

Необходимо выключить запись в архив СПАД при возникновении 10 ошибок записи на диск подряд. Включение записи в архив осуществляется только по команде оператора.

Для решения этой задачи нужно два канала. Первый из них должен иметь тип **INPUT**, подтип **ДИАГНОСТИКА** и дополнение к подтипу [СПАД](#). Равенство его значения 3 является индикатором ошибки записи. Зададим для него имя СТАТУС.

Второй канал предназначен для выключения записи в СПАД. Зададим ему тип **OUTPUT**, подтип **СИСТЕМНЫЙ** и дополнение [Архивация](#). Имя данному каналу дадим УПР\_СПАД. Для этого канала надо настроить процедуру управления.

Создадим FBD-программу. Она должна подсчитывать количество тактов, в течение которых значение первого аргумента было равно 3. При равенстве накопленного значения 10, второй и третий аргументы должны быть установлены в 1. Имя этой программе установим УПР\_СПАД. Далее настроим процедуру УПРАВЛЕНИЕ для канала УПР\_СПАД. Выберем для нее соответствующую FBD-программу. Затем в качестве первого аргумента укажем реальное значение канала СТАТУС, в качестве второго - первый бит канала УПР\_СПАД, а в качестве третьего - состояние канала УПР\_СПАД.

На этом решение данной задачи закончено. Теперь при неисправности жесткого диска после десяти неудачных попыток сохранения архивирование данных будет остановлено.

### ***Синхронизация локальных архивов***

Для дублированных мониторов реального времени (**Double Force MPB**) реализована поддержка синхронизации локальных архивов.

Для управления синхронизацией используется канал типа **OUTPUT** подтипа **ДИАГНОСТИКА** с дополнением **СПАД синхронизация**. Посылаемые в него значения задают следующие команды:

- 0 - ничего не делать;
- 1 - синхронизировать сейчас;
- 2 - отключить автоматическую синхронизацию;
- 3..64000 - установить интервал синхронизации в секундах.

По умолчанию период синхронизации равен 60 секунд.

**Внимание!** Если этот канал имеет тип INPUT, то он контролирует текущее состояние архивирования.

Поддержка автоматической синхронизации и имени компьютера с резервным MPB задаются в реестре Windows:

#### **Ключ:**

HKEY\_LOCAL\_MACHINE\SOFTWARE\AdAstra\TRACE MODE\5.0\SIAD

#### **Значения:**

"AutoSync"=dword:00000001

1 – включить автоматическую синхронизацию;

0 – отключить;

"ServerName"="имя компьютера"

"имя компьютера" – имя компьютера, где запущен резервный MPB.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 8. Языки разработки алгоритмов Trace Mode 5.** Справочная система Trace Mode 5, раздел: «Языки разработки алгоритмов», глава: «Язык функциональных блоков»:

- программирование блоков на Техно II;
- создание блока из FBD-программы;
- подключение внешних алгоритмов.

глава «Язык инструкций».



## ЛЕКЦИЯ 9

# СИСТЕМЫ УПРАВЛЕНИЯ ПРОИЗВОДСТВЕННОЙ ИНФОРМАЦИЕЙ

1. Общие характеристики производителей PDM
2. Функциональные возможности PDM
3. Пользовательская среда
4. Операционная среда
5. Оценки перспективности различных систем PDM
6. Архивирование в Trace Mode 5
  - отчёт тревог;
  - глобальный регистратор.

*Наталья Дубова* [os@osp.msk.su](mailto:os@osp.msk.su)

[Общие характеристики производителей PDM](#)

[Функциональные возможности PDM](#)

[Управление хранением данных и документами](#)

[Управление потоками заданий и процессами](#)

[Управление структурой продукта](#)

[Другие возможности](#)

[Пользовательская среда](#)

[Операционная среда](#)

[Оценки перспективности различных систем PDM](#)

[Литература](#)

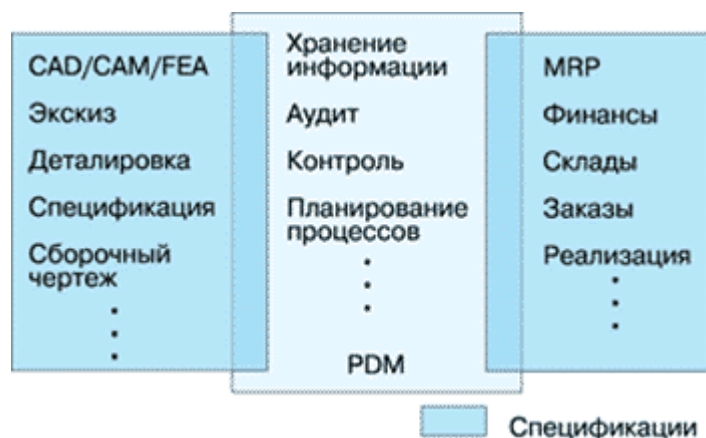
---

*Проблема отслеживания процесса продвижения информации и управления документами на производстве, в условиях современного крупного промышленного предприятия сегодня приобретает особую остроту. До недавнего времени ведущие компании - производители программного обеспечения радовали промышленников в лучшем случае системами САПР с элементами аналитических программ. Однако, инженеры крупных предприятий, первоначально освоив САПР как средство повышения комфорта при проектировании новых изделий, "попутно" значительно повысили производительность своего труда. Оказалось, что теперь, даже простое сопровождение новых проектов с отслеживанием изменений и регистрацией утверждений без автоматизации становится тяжелой задачей. "Доморощенные" подделки типа многочисленных и избыточных ошибок интерфейсов к СУБД на ПК не только не решили задач автоматизации управления технологическими данными предприятий, но и породили у капитанов индустрии стойкий синдром неприятия любых новшеств в этой области. Время тем не менее уже призвало к ответу разработчиков и последние не заставили себя долго ждать - появилась целая плеяда развитых систем управления производственной информацией или PDM (Product Data Management).*

Косвенным доказательством растущей активности производителей в области PDM является и существенное увеличение всевозможных обзоров и конференций по этой тематике [1-5]. Наиболее значительными конференциями в прошлом году стали: Kalthoff Group EDMS Conference (Чикаго, США) и европейский форум американского консультационного союза CIMdata - PDM Europe'95 (Новдвийк, Нидерланды). Отдельным

изданием стал выходить журнал EDM Report, посвященный тематике PDM и кроме того рейтинговыми исследованиями PDM-систем занялись такие ведущие аналитические центры как Gartner Group, D.H.Brown Associates, Dataquest и др. PDM - относительно новая технология, и ее рынок сегодня быстро развивается. По оценкам CIMdata, объем продаж систем PDM возрастет с 450 млн. долл. в 1994 году до 1.2 млрд. в 1998 году. Основными потребителями этого класса продуктов, а значит и основными движущими силами развития рынка PDM являются такие отрасли, как аэрокосмическая и автомобильная промышленности и электроника, хотя сейчас к ним присоединяются и представители других направлений, например, энергетические компании.

Прежде чем перейти к анализу продуктов, предлагаемых различными производителями определим место систем PDM в общей производственной цепочке. На рис. 1 представлена схема взаимоотношения систем CAD/CAM/ FEA, PDM и MRP.



**Рисунок 1.**  
Взаимосвязь систем автоматизации производственных процессов.

Системы PDM играют роль связующего звена между этапом инженерно-конструкторской подготовки нового изделия и системами MRP (Manufacturing Resource Planning) или, по простому, разного рода АСУ, решающих задачи автоматизации управления финансами, складским хозяйством, снабжением и сбытом, а также техническим обслуживанием. Типичными системами группы CAD можно назвать Pro/Engineer, Euclid, Anvil, Cosmos/M, а группы MRP - R3 или Manman/X компании Computer Associates. Находясь между условными входами и выходами корпорации, системы PDM аккумулируют все циркулирующие внутри компании данные по продукции, осуществляют планирование процессов и пошаговый контроль - иначе говоря, система PDM является идеальным рабочим местом руководителя проекта. О важности такого рода систем свидетельствует хотя бы такой известный факт, что только 25% рабочего времени персонала компании, начиная от проектировщика и кончая руководителем проекта тратиться на собственно творческую работу, остальное - это поиск информации и стыковка потоков данных, поступающих от разных подразделений корпорации. Часто оказывается, что проще заново разработать деталь, чем найти информацию по ней, подготовленную некоторое время назад.

### **Общие характеристики производителей PDM**

Системы PDM разрабатываются различными компаниями, которые одновременно занимаются выпуском таких продуктов, как САПР или обработки изображений. Несмотря на большое количество альтернативных поставщиков, CIMdata не выделяет какого-то абсолютного лидера рынка PDM, вследствие его неразвитости и присоединения к нему

все новых и новых участников. Тем не менее в тройку лидеров по продажам продуктов PDM за последние три года вошли Computevision, Hewlett-Packard и Sherpa. За ними следуют SDRC, Intergraph, Eigner+Partner, DEC, Control Data, IBM, Workgroup, Metaphase и EDS. В этом обзоре основное внимание будет уделено продуктам трех ведущих фирм, а также системам компаний Metaphase, Adra, Eigner+Partner и EDS, поскольку в концептуальном плане они представляют наибольший интерес.

· Компания Sherpa (Сан Хосе, шт. Калифорния) специализируется на разработке систем PDM и занимает одно из ведущих положений на этом рынке. Первый продукт - DMS, реализующий возможности электронных хранилищ данных, компания выпустила в 1986 году. Несколько лет назад компания разработала серию PDM-приложений под названием PIMS, и сегодня все продажи Sherpa на рынке PDM составляет комбинация DMS и PIMS.

Система DMS/PIMS завоевала уже большую популярность среди пользователей. Это надежный продукт, обеспечивающий сбалансированную комбинацию основных функциональных возможностей, системной архитектуры и удобного пользовательского интерфейса. Стратегией Sherpa в последние годы было обеспечение хороших возможностей по всем направлениям, присущим системам PDM, а не реализация лучшего из лучших в какой-то одной функциональной области. Большое внимание при работе над своей системой компания уделяла также простоте использования и понимания с точки зрения конечных пользователей. Эта философия простого пользовательского интерфейса роднит Sherpa с такими компаниями, как HP и Metaphase, хотя их системы во многом отличаются друг от друга.

DMS/PIMS представляет собой законченное приложение PDM, которое относительно просто настроить на решение задач конкретных пользователей. Подобный же подход реализуют в своих продуктах компании Metaphase, Eigner и HP. Другая возможность - предоставить пользователю пакет инструментальных средств для реализации приложений PDM, как это сделано в системе Matrix компании Adra или в Digital OpenDATA Manager. Архитектура DMS/PIMS способна поддерживать крупномасштабные реализации, и на сегодняшний день самые большие реализации PDM принадлежат именно компании Sherpa.

Sherpa прилагает значительные усилия, для того чтобы ее продукт мог быстро реализовываться с минимальными затратами на сопровождение. Sherpa - небольшая компания, поэтому в ее интересах снизить затраты на поддержку, чтобы высвободить силы на быструю разработку новых программных систем.

· Computervision (CV, <http://www.cv.com/>) - хорошо известный во всем мире разработчик систем CAD/CAM. Самые первые возможности управления данными компания включила в свой пакет CADDs еще в 1984 году, а через пять лет была выпущена PDM-система под названием EDM, новая версия которой (1995 год) получила название - Optegra. Смена названия кроме всего прочего вызвана еще и тем, что CIMdata зарезервировала сокращение EDM (Enterprise Data Management - системы управления техническими данными в рамках корпорации) за всем классом продуктов и намерена впредь в своих материалах использовать его вместо устаревшего PDM.

Optegra является комбинацией объектно-ориентированной технологии и возможностей, реализованных в старом продукте EDM. При продажах EDM компания опиралась главным образом на клиентов своих CAD/CAM-систем и добилась здесь больших успехов. Однако сейчас CV особенно заинтересована в расширении своего влияния за пределы инсталляционной базы CADDs. Optegra является набором функциональных

модулей, которые в совокупности способны реализовать всю палитру функциональных возможностей PDM-системы. Однако стратегия CV в отношении Optegra заключается в предоставлении клиентам готовых программ, ориентированных на поддержку определенных решений, в которых комбинируются программные модули, сервисы и т.д. Например, Total Data Management (TDM) - программа общего управления данными, Configuration Management (CM) - программа управления конфигурацией. Такой подход позволяет решать конкретные задачи рынка и поддерживается специальной маркетинговой программой и программой обучения конечных пользователей. Отличительной чертой Optegra являются способы ее интеграции с наиболее распространенными САПР: CADDSS, MEDUSA, Dimension-III, CATIA (IBM/Dassault), Pro/Engineer (PTC), AutoCAD (Autodesk), Design P&D (CV).

По данным CIMdata, Computervision принадлежит максимальная доля на рынке PDM. Успех компании во многом объясняется ее большим опытом в разработке САПР и большой инсталляционной базой в этой области. Система Optegra имеет хороший потенциал стать лидером в этой области, особенно в реализациях PDM в инженерных организациях, для которых необходим хороший интерфейс с механическими CAD-системами.

Недавно компания начала реализацию новой программы для работы со своими клиентами, которая вовлекает основных заказчиков компании в процесс разработки спецификаций и усовершенствований для Optegra. Тогда же фирмами CV и SAP было объявлено о совместной разработке программно-функционального интерфейса между системами Optegra и R/3. В России к началу 1996 года было произведено несколько инсталляций системы. Наиболее значительные - на Ульяновском Автозаводе, на Нижнетагильском Металлургическом Комбинате и в КБ Сухого. Большой опыт по локализации системы накоплен также программистами УАЗа.

· Компания Hewlett-Packard не нуждается в специальных представлениях, а ее система WorkManager хотя и появилась на рынке PDM сравнительно недавно, но уже успела приобрести хорошую репутацию. В 1993 году HP образовала специальную группу для работы над WorkManager. На рынке это было расценено как знак особого внимания HP к технологии PDM и, в частности, к своей системе, которая может стать одним из основных направлений деятельности компании в долгосрочной перспективе.

WorkManager - это гибкая система PDM, которая легко может настраиваться на решение задач конкретных пользователей. Репутация HP, ее опыт и стабильность в бизнесе, как и возможности разработанной системы дают компании большие шансы на успех на этом рынке. Однако у WorkManager есть и функциональные недостатки и, кроме того, надо отметить отсутствие хорошо подготовленных групп по продажам и поддержке, таких, например, как у Computervision. Тем не менее, WorkManager получает большой кредит доверия от своих потенциальных пользователей на преодоление имеющихся недостатков и создание новых возможностей.

· Metaphase - относительно новая компания в индустрии PDM. Она была образована в 1992 году как совместное предприятие корпорации Control Data Systems (CDS) и Structural Dynamic Research Corporation (SDRC). Деятельность этих двух компаний по разработке и маркетингу продуктов PDM была объединена с целью созданию новой передовой технологии в этом направлении. Системы EDL от CDS и DMCS от SDRC занимали ведущие позиции на рынке PDM, обладая при этом различными характеристиками и дополняющими друг друга сильными и слабыми сторонами. Новая компания должна была

создать продукт, объединяющий и расширяющий возможности своих предшественников. Первой версией такого продукта является система Metaphase 2.0.

Metaphase 2.0, как и системы компаний Sherpa, Eigner и HP, является интегрированным PDM-приложением, которое легко настраивается на выполнение задач конкретных пользователей. При этом Metaphase обеспечивает возможности расширения с помощью дополнительных приложений. Metaphase 2.0 реализует хорошие возможности по всем основным направлениям PDM и один из лучших пользовательских интерфейсов в этой области.

Бизнес компании Metaphase строится не на прямых поставках ее продукта конечным пользователям, а на создании системы связей с другими компаниями, которые используют Metaphase как базовую технологию PDM, дополняют ее своим собственным опытом, возможностями интеграции с другими приложениями и приложениями расширения, после чего выпускают продукт на рынок. Дистрибьютерами Metaphase являются прежде всего ее "родители" - CDS и SDRC - а также многие другие компании, список которых продолжает расширяться. За последнее время Metaphase установила партнерские связи с компаниями Bull, Siemens Nixdorf, FORMTEK и Intergraph.

- Компания Adra Systems образовала подразделение Matrix по разработке системы PDM в 1992 году. Продукт с тем же самым названием вызывает особый интерес на рынке, поскольку является первой реализацией PDM на основе исключительно объектно-ориентированной СУБД. Отсюда вытекают как сильные, так и слабые стороны этой системы. Благодаря объектно-ориентированному стилю, а также очень удобному пользовательскому интерфейсу система Matrix хорошо настраивается на задачи конкретных пользователей, очень проста в использовании, может легко изменяться и модифицироваться в случае необходимости. Однако использование ООСУБД вызывает не только интерес, но и определенную настороженность со стороны компаний, привыкших работать с более традиционными базами данных. Кроме того, система Matrix - это не стандартное PDM-приложение, а скорее набор возможностей, которые могут использоваться для построения таких приложений. Это удобно для пользователей, нуждающихся в среде для экспериментов и испытания новых идей по управлению данными и процессами. Но такой продукт может быть негативно встречен организациями, требующими стандартных приложений для работы со своей информацией.

- Германская компания Eigner+Partner, основанная в 1985 году, является основным поставщиком продуктов PDM в Центральной Европе. Ее система CADIM/EDB - это стандартное PDM-приложение, которое легко настраивается на решение задач конкретных пользователей с помощью специальных таблиц, правил и т.д. Разработанная на языке четвертого поколения Dataview, CADIM/EDB является полностью открытым для пользователей продуктом, в который можно легко вносить изменения. Eigner уделяет особое внимание поддержке клиентских версий своей системы - разработана серия средств для анализа клиентских кодов, которые используются для поддержки новых версий CADIM/EDB.

CADIM/EDB отражает обширный опыт Eigner в области механических САД-систем, и была особенно хорошо принята в инженерной среде. Компания затратила много усилий на разработку возможностей интеграции с многочисленными САД-системами, первоначально преимущественно механическими, а затем и электронными. Особое значение этих возможностей не в их количестве, а в том, что Eigner разработала механизм их поддержки как коммерческих продуктов.

· Фирма EDS Unigraphics (Мэриленд Хайтс, США <http://www.ug.eds.com/>) - дочернее предприятия корпорации General Motors - одна из ведущих компаний в области САПР в мире. В прошлом году EDS Unigraphics (UG) анонсировала новую версию своего продукта класса PDM - IMAN V3.2, представляющего собой комплексное решение по автоматизации документооборота корпорации. IMAN V3.2 содержит большое количество новых возможностей, особенно для конфигурирования и управления документами и прикладными программами. Для навигации по структуре продукта используется модуль - Advanced Product Structure Manager (PSM) с улучшенным пользовательским интерфейсом, позволяющим осуществлять установку на разные платформы, как UNIX станции, так и ПК. Модуль системного администратора System Administration (SA) облегчает работы по генерации/откатки/восстановлению системы, конфигурации пользователей и установке прав доступа к информационным компонентам базы данных, обеспечении взаимодействия с базовой реляционной СУБД Oracle. В версию IMAN V3.2 добавлен модуль обработчика потока заданий IMAN Workflow. В России компания EDS работает уже с 1992 года и сегодня ее продукты можно встретить на АвтоВАЗ, Казанском Вертолетном заводе, КБ Сухого, СТИНОЛ и др.

## **Функциональные возможности PDM**

Функциональные возможности систем PDM охватывают несколько направлений, среди которых основными являются организация хранения данных и управление документами, управление потоком работ и процессами, управление структурой продукта, автоматизация генерации выборок и отчетов.

### **Управление хранением данных и документами**

Во всех рассмотренных системах реализован сходный набор средств организации хранения данных и управления документами: возможности электронных хранилищ данных (в литературе о PDM для их обозначения принят термин data vault), управление уровнями версий, контроль авторизации для защиты доступа к информации. Например, в системе Optegra (CV) функциональный модуль управления хранением данных Optegra Vault, являющийся ядром всей системы, включает в себя интегрированную систему электронной почты, распределенное по сети хранение данных и управление файлами, контроль защиты/доступа, резервирование/восстановление, генерацию сообщений и возможности архивирования.

Функции управления хранением позволяют определять различные ревизии частей/элементов данных и отношения между частями и элементами (или документами), которые определяют эти части. Легко и быстро могут создаваться новые типы объектов, которые наследуют атрибуты и связанные с ними действия или процессы объектов-родителей. Такие объекты-потомки в свою очередь могут получать дополнительные атрибуты и процессы, которые определяются специально для них. Например, в системе Matrix (Adra) эти действия выполняются особенно просто.

Механизм авторизации для защиты данных в системах PDM позволяет ограничить доступ, определяя права отдельных пользователей или их групп, а также по статусу определенной части данных. Например, система CADIM/EDB (Eigner+Partner) обеспечивает множество прав и пользовательских привилегий, по которым определяется, кто может осуществлять доступ, изменять и удалять информацию. Кроме того, могут быть заданы пользовательские привилегии, определяющие, какие команды может выполнять данный пользователь.

Для самих хранилищ данных продукты PDM обеспечивают одновременно отказоустойчивость и простоту работы с информацией. К ним относятся функции "check-in" и "check-out" (первичное размещение данных в управляемой области хранения и их окончательное восстановление из хранилища), хранение и выборка объектов (документов), доступ к файлам посредством программируемого интерфейса и др.

В системе CADIM/EDB (Eigent) управление документами сосредоточено на объектах с гибкими и настраиваемыми на потребности пользователей подклассами (тип документа). Файлы и объекты находятся в соотношении 1 : n. Объект может включать любой файл с цифровой информацией, полученной и созданной различными прикладными системами, и может содержать ссылки на документы, которые хранятся в обычном виде на бумаге. CADIM/EDB поддерживает задание множества отношений между документами и определение "объектов" более высокого уровня, которые могут использоваться для управления скоординированными группами файлов.

Практически все коммерческие системы PDM игнорируют пока возможности управления прикладными средствами создания/модификации документов. Однако компании Computervision, Metaphase, а также IBM начинают включать в свои продукты такие средства. Система Metaphase может управлять версиями прикладного программного обеспечения, которое используется для создания файлов, что позволяет гарантировать согласованное применение пользовательских инструментальных средств.

Metaphase поддерживает еще одну, относительно уникальную возможность управления хранением. В большинстве систем PDM элемент данных, за управление которым отвечает система, должен быть защищен в электронном хранилище. Metaphase 2.0 позволяет управлять как данными, которые находятся в хранилище, так и данными, которые там не размещены. Конечно, данные, отсутствующие в хранилище, не имеют средств защиты, но зато пользователи получают более гибкие возможности работы с ними. Они сами определяют, какие данные должны быть защищены, а какие нет. Если пользователи хотят надежно хранить все данные, они определяют соответствующее правило, которое и выполнит система.

В области управления хранением документами интерес представляет также возможность хранения как текстовых, так и графических документов. Для текстовых документов существует свое множество функций поиска, выборки и отображения, например, выборка полного текста (full-text-retrieval - FTR), задание структурной архитектуры документа и т.д. Система Optegra (CV) реализует возможности FTR и SGML, что дает ей определенные преимущества перед продуктами других компаний, в особенности для тех клиентов, в корпорациях которых технология PDM реализуется достаточно широко.

### **Управление потоками заданий и процессами**

Управление изменениями - это функциональная область, в разработку которой компании, производящие продукты PDM, вкладывают самые значительные усилия. Одновременно, это область наибольших потенциальных различий между системами PDM. Поставщики продуктов PDM стремятся предоставить возможности управления потоками заданий и процессами в виде стандартных функциональных модулей. Все большее значение уделяется графике как средству определения и управления потоками и процессами.

Определение процесса изменений - это важная часть управления изменениями. Сюда относится определение упорядоченных этапов процесса, правила, связываемые с этими этапами и правила для подтверждения каждого этапа. Рассматриваемые системы PDM



обеспечивают в основном сходные функции в этой области: поддержка нескольких типов параллельных и последовательных процессов, условные переходы, параметры синхронизации и др. По оценкам CIMdata, наибольшей гибкостью и здесь отличаются система компаний Metaphase, а также ProductManager от IBM.

Функциональный модуль системы Optegra (CV), отвечающий за управление потоками заданий и процессами - Optegra Workflow- представляет собой настольную среду типа Microsoft Office, использующую папки (фолдеры) для хранения заданий и доступных данных. Концепция фолдеров при управлении потоками применяется и в других системах PDM. Optegra Workflow использует несколько различных типов заранее определенных элементов потоков заданий. Пользователям предоставляется возможность графического представления потоков заданий и процессов, а также создания потоков графических элементов. Подобные функции предоставляются или разрабатываются всеми основными поставщиками PDM. Отличительной чертой Optegra Workflow является - это тесная интеграция с модулем управления хранением и документами - Optegra Vault, а также замкнутый характер этого автономного приложения, что позволяет Computervision независимо продвигать на рынок этот модуль для различных приложений, причем не только в традиционной инженерной области.

Система WorkManager (HP), в отличие от других, не обеспечивает заранее разработанного приложения для формального автоматического определения потоков заданий и процессов. В принципе такое определение возможно, но для этого необходимо специальное кодирование на языке макропрограммирования WorkManager. Эта система поддерживает концепцию потоков заданий ad hoc, для которых не дается строгого, формализованного, заранее точно описанного определения условий перехода. Такие возможности также представляют определенный интерес, и сейчас многие поставщики PDM работают над их реализацией в своих системах. Эти особенности WorkManager лучше подходят инженерным группам, где не требуется формального описания процессов и где актуальной является задача компьютерной поддержки поисковых и исследовательских задач.

В системе CADIM/EDB (Eigner) предусмотрено различие между процессом и потоком заданий. Процесс - это множество состояний и переходов вместе с дополнительными функциями и правилами, которое настраивается на нужды пользователей, но является стабильным на время выполнения. Процесс может быть определен системным администратором и может соединяться с каждым объектом и документом. Поток заданий - это временный процесс, определяемый конечным пользователем. Концепция потока заданий в CADIM/EDB аналогична функциям потоков ad hoc в WorkManager.

## **Управление структурой продукта**

При решении задач управления структурой продукта используется наглядный и ясный подход к отображению сложного изделия в виде иерархического дерева отношений, типа "деталь-сборка-агрегат-изделие". При таком подходе корень дерева структуры - это собственно имя изделия, а концевые листья - конкретные детали, составляющие это изделие. Компонентное наполнение подобной структуры может быть различным и разнотипным - текстовый файл, бинарный файл, файл пространственной модели, атрибут и т.д. Функциональная область включает в себя средства взаимодействия, манипуляции, создания и модификации структуры продукта и управление конфигурацией. Так, например, прикладной модуль Optegra Navigator системы Optegra (CV) реализует графический пользовательский интерфейс, который обеспечивает доступ к структурам продуктов, управляемых и поддерживаемых Optegra. Графические возможности



просмотра и отображения структуры продуктов реализуют в той или иной степени все рассматриваемые системы PDM, но наиболее сильные из них принадлежат компаниям Sherpa, Metaphase и Computervision. По оценкам CIMdata, пользовательский графический интерфейс системы Metaphase 2.0 - лучший среди продуктов этого класса.

Предшественник Optegra, система EDM не имела эффективных способов редактирования структуры продукта и была зависима в этом отношении от CAD-системы компании Computervision - CADDs. Optegra ликвидирует этот недостаток, предоставляя независимые от CADDs возможности определения и модификации структуры продукта. При этом, как уже отмечалось, остается тесная интеграция между CADDs и Optegra.

Большинство систем PDM обеспечивают эквивалентные базовые возможности манипулирования структурой продукта. Сюда относится определение и модификация структуры, поддержка версий и опций дизайна и другие возможности. Несмотря на сходство спектра предоставляемых средств, в этой области существуют реальные различия между системами, которые относятся к типам отношений, поддерживаемым в структуре, и механизмам, которые используются для реализации опций и версий. В системе Matrix (Arda) есть средства построения структуры продукта, но не реализованы предварительно определенные элементы структуры и отношения, как это сделано в других системах.

В системе Metaphase 2.0 реализован специальный модуль управления конфигурацией, который объединяет управление потоками заданий и процессами и управление структурой продукта. Расширенные функции этого модуля обеспечивают улучшенный контроль за процессом, множество представлений структуры, анализ влияния изменений, заменяющие и альтернативные элементы, а также другие возможности. Подобные средства реализованы и в модуле управления конфигурацией системы Optegra. WorkManager (HP), как уже отмечалось, не имеет интегрированного модуля управления изменениями, и это мешает реализовать в этой системе полный спектр возможностей управления конфигурацией.

Концепция множества представлений структуры продукта или конфигурации, реализованная, в частности, в системах компаний Metaphase, Computervision, Sherpa и IBM, представляет особый интерес для производящих компаний, поскольку предоставляет хорошую возможность разработки интерфейсов с системами планирования распределения ресурсов (MRP-системами). Тесная связь между системами PDM и MRP становится особенно актуальной для компаний, которые занимаются реорганизацией своего бизнеса. Интересно, что по оценкам CIMdata интеграция инженерных и производственных систем имеет наибольший приоритет именно в странах Европы.

Управление структурой продукта предоставляет возможности и для разработки интерфейсов с другими прикладными системами. Уже отмечалась тесная связь EDM и Optegra с системой CADDs компании Computervision. Элементы связи со своими CAD-системами поддерживают компании Hewlett-Packard, IBM, Eigner+Partner. И большинство поставщиков PDM-систем также задумываются над реализацией таких возможностей.

## **Другие возможности**

Технология PDM предусматривает реализацию средств классификации данных о продукте. Большинство систем PDM предоставляют множество средств для определения и использования атрибутов частей и других объектов в системе. Различия между продуктами в этой области обычно относятся к гибкости определения атрибутов,

поддержке фолдеров, структуре классификации и предоставляемым функциям поиска и запроса. Система Optegra, например, не реализует специального приложения классификации, но в качестве стандартных возможностей предоставляет средства определения атрибутов и множество правил классификации, по которым атрибуты присваиваются определенной части.

Определение атрибутов - это только один элемент классификации. Системы компаний Eigner, Metaphase и HP предоставляют более существенные возможности, обеспечивая механизм определения и управления иерархическими схемами классификации и соотношения частей/элементов с узлами. Механизм иерархической классификации может оказать реальную помощь в использовании стандартных частей, процессов, пакетов и т.д.

Важной задачей для продуктов PDM является обеспечение интерфейса с системами управления проектом, так чтобы накопленная информация, которая содержится в PDM-системе, могла эффективно использоваться для поддержки управления проектом. Большинство систем PDM не предоставляют сегодня непосредственной поддержки таких возможностей, за исключением продукта компании Sherpa, в котором реализованы средства создания иерархии задач и подзадач, представляющих структуру проекта. Эти задачи используются в качестве основных точек связи между DMS/PIMS и системами управления проектом.

Для поддержки внутренних коммуникаций рассматриваемые системы используют либо свою собственную почтовую систему, как, например, Optegra, либо систему E-mail заказчика.

Все большую важность в системах PDM приобретают службы представления модели продукта в виде графических изображений, которые хранятся как элементы системы. Для преобразования этих изображений из различных форматов в представимую форму используются различные типы схем, работающие либо в реальном времени, либо в соответствии с некоторыми заранее определенными событиями. Однако большинство этих схем основывается на использовании растровых или векторных форматов для представления на UNIX-станциях или ПК.

Раньше подобные возможности предоставлялись в PDM-системах как часть отдельной, связанной системы вывода изображений. Однако за последнее время значительно повысилась заинтересованность пользователей в средствах работы с изображениями, представляющих собой составную часть общего процесса наблюдения за изменениями. Поэтому сейчас большинство поставщиков продуктов PDM обеспечивают эти возможности как интегральную часть своих систем, используя для этого различные прикладные пакеты других компаний.

## **Пользовательская среда**

Понятие пользовательской среды PDM включает в себя стиль интерфейса пользователя, простоту работы с системой, настраиваемость интерфейса на нужды пользователя и предоставляемые системой прикладные интерфейсы.

Все основные участники рынка PDM сегодня имеют пользовательские интерфейсы типа X-Window (Motif или Open Look) и MS-Windows. Графический стиль взаимодействия с пользователями теперь уже не отличительная черта, а скорее основное требование к конкурентноспособной системе. Такие компании, как Computervision, Metaphase, Sherpa и Eigner выбрали в качестве базы коммерческий продукт - GUI компании Neuron Data. Одна

из основных характеристик графического интерфейса системы - так называемая "look and feel" - выразительность, интуитивная ясность способа взаимодействия с пользователем. CIMdata очень высоко оценивает это качество у системы Ortegra компании Computervision. Отмечается также легкость освоения интерфейса Ortegra для неопытных пользователей.

Один из лучших среди систем PDM графических интерфейсов предоставляет компания Metaphase - для управления структурой продукта предлагаются функции редактирования на месте (edit-in-place), которые пока отсутствуют у альтернативных продуктов других производителей.

Хорошие характеристики по простоте использования имеют системы компаний HP, Sherpa, Metaphase, Computervision. Но особую работу в этом направлении проделала Sherpa, которая, сосредоточив особое внимание на интересах своих конечных пользователей, реализует их запросы в серии приложений PIMS. Очень удобный, функциональный графический пользовательский интерфейс предоставляет компания Adra в своей системе Matrix. Как уже отмечалось, объектно-ориентированная направленность делает эту систему достаточно гибкой, легко настраиваемой на задачи конкретных пользователей и простой в реализации.

Простота использования системы PDM определяется тем, с какой точки зрения пользователи смотрят на эту систему. Для одних она служит базой для доступа к файлам, информации и запуска специальных приложений. Другие используют систему PDM непосредственно из определенных приложений. Для первой категории пользователей более удобен интерфейс, имеющийся в портативных компьютерах Mac и MS-Windows и обеспечивающий простые, основанные на использовании меню действия для доступа к данным, информации и приложениям. Вторая категория пользователей предпочитает, чтобы функции PDM были доступны из их приложения с помощью обычного для данного приложения интерфейса. Для этих целей системы PDM обеспечивают прикладные интерфейсы, которые являются еще одним компонентом пользовательской среды.

Так, системы Ortegra на сегодняшний день поддерживает прикладные интерфейсы с пакетами CADDs, Pro/ENGINEER, DesignPost P&D, AutoCAD, EpochBackup, Legato Networker и планирует реализацию интерфейсов с системами CATIA, MEDUSA и SAP. Доступ ко всем поддерживаемым системой API обеспечивает функциональный модуль Ortegra Customizer. Это означает, что любая возможность Ortegra может быть использована соответствующим приложением. Ряд других поставщиков продуктов PDM обеспечивают аналогичные возможности, например, Metaphase, Sherpa и HP предоставляют обширные библиотеки для доступа к своим PDM-функциям.

Большинство пользовательских организаций хотят иметь не только хороший набор базовых интерфейсов с прикладными системами, но и пакет инструментальных средств для разработки дополнительных интерфейсов. Здесь можно выделить деятельность компании Eigner, которая предоставляет своим клиентам и партнерам средство разработки DataView, на котором написана и сама система PDM этой фирмы. Eigner разработала специальное расширение, которое позволяет разрабатывать интерфейсы с механическими САД. Уже отмечалось, что отличительной чертой интерфейсов, поддерживаемых Eigner, является то, что они приняты как промышленные стандарты. Eigner реализует интерфейсы с системами UG, ME10/30, Euclid (Matra Datavision), Pro/Engineer, CATIA и многими другими.

Еще один аспект пользовательской среды - это возможность ее настройки на задачи пользователей. Опрос, проведенный CIMdata среди пользователей систем PDM, показал, что они рассматривают способность такой настройки в качестве ключевого элемента успешной реализации PDM.

Большинство систем поддерживают подобные возможности, правда обычно они предназначены для использования администраторами систем. Metaphase и Sherpa поддерживают множество версий пользовательской среды. В системе компании HP несколько интерфейсов могут создаваться различными пользователями, настройка системы может производиться в соответствии с правами авторизации, определенными системой. Возможности, предоставляемые Eigner, требуют от пользователей, выполняющих изменения большего опыта и обширных знаний о системе. Computervision обеспечивает полный набор средств настройки системы в серии модулей Optegra Customizer.

## **Операционная среда**

Продукты всех перечисленных фирм реализуют распределенную модель клиент/сервер и на основе стека сетевых протоколов TCP/IP поддерживают гетерогенные платформы как на стороне клиента, так и на стороне сервера. Все компании используют UNIX-системы от различных поставщиков. Optegra поддерживает реализации UNIX от HP, Sun, IBM, Digital и Silicon Graphics. Sherpa, Metaphase и Eigner также предлагают широкий диапазон UNIX-платформ и обеспечивают поддержку VAX/VMS, которой нет в продуктах HP и Computervision.

Сегодня большой интерес вызывает ОС Windows NT. Большинство поставщиков PDM не спешили с переносом своих систем на NT, пока не было полной ясности, как этот продукт будет принят на рынке. Computervision реализовала на NT как серверную, так и клиентскую версии своей системы, и в этом опережает пока своих конкурентов. Поддержку сервера на Windows NT в прошлом году предоставила также Metaphase. Ожидается, что и другие поставщики последуют примеру этих компаний.

Особое внимание уделяется ПК IBM PC и Macintosh, поскольку эти платформы имеют широкую инсталляционную базу и активно применяются конечными пользователями. До недавнего времени клиентские версии систем PDM на этих платформах были не более чем использованием ПК в качестве X-терминалов или чаще - простых терминалов исходных данных. Сейчас это уже не может удовлетворить требований рынка и большинство систем реализуют определенные возможности для поддержки ПК как терминалов клиентов с их собственным интерфейсом: MS-Windows или MacOS. Система Optegra поддерживает полноценных клиентов на обеих этих платформах, и в этом также является лидером.

Большинство производителей систем PDM не поддерживают и не планируют поддерживать сегодня мейнфреймы IBM, за исключением самой IBM. Они рассчитывают только на интерфейсы с приложениям класса PDM, которые выполняются на машинах этого типа. Основным среди таких приложений является IBM ProductManager.

Для работы с базами данных большинство систем PDM реализуют так называемый "нейтральный SQL-уровень" между кодом приложения и СУБД. Затем разрабатывается специальный интерфейс, который адаптирует нейтральный SQL к определенной системе. Такой подход упрощает настройку на работу с дополнительными базами данных.

Основные системы PDM базируются сегодня на реляционных СУБД, лидирующей среди которых является продукт фирмы Oracle. Но доступны и другие системы. Так, например, Eigner помимо Oracle поддерживает базы данных Ingres, RDB и INFORMIX. Metaphase помимо Oracle планирует поддержку Sybase и INFORMIX. Система Optegra работает только с Oracle. Matrix - это единственный коммерческий продукт PDM, поддерживающий объектно-ориентированную СУБД: Objectivity. Однако архитектурные принципы систем компаний Metaphase и Computervision также позволят им использовать ООСУБД в будущем.

Поддержка распределенных хранилищ данных является одним из основных требований к реализации систем PDM в масштабах предприятия. Возможности работы с неограниченным числом распределенных баз данных и поддержку множества конфигураций серверов баз данных предоставляют сегодня системы Computervision, Metaphase, Sherpa и IBM. Eigner и HP не предлагают таких возможностей, однако Hewlett-Packard анонсировала их реализацию.

### **Оценки перспективности различных систем PDM**

Рейтинги среди систем PDM распространяются независимыми консультационными фирмами и обычно оценки в таких рейтингах носят не количественный, а качественный характер и часто не совпадают у разных фирм. Проводя интегральную оценку рейтингов по ведущим изданиям в 1995 году, можно разнести эти оценки по трем направлениям: поставки программных продуктов (SW), продажи комплексные - программы и сервис по поддержке пользователя программ (SW+SRV), поставки готовых решений в промышленность (SW4IND). Результаты анализа публикаций компаний CIMdata, Gartner Group, Dataquest, D.H.Brown Associates, IDC собраны в таблице 1.

<b>Таблица 1.</b>			
<b>Название фирмы</b>	<b>Оценка применимости</b>		
	<b>SW</b>	<b>SW+SRV</b>	<b>SW4IND</b>
<b>Computervision</b>	отл	отл	отл
<b>Sherpa</b>	удовл	отл	хор
<b>SDRC (Metaphase)</b>	удовл	отл	хор
<b>Hewlett-Packard</b>	хор	хор	отл
<b>Adra Systems</b>	удовл	удовл	хор
<b>Eigner+Partner</b>	удовл	удовл	хор
<b>EDS UG</b>	хор	хор	отл

В качестве дополнения можно привести сравнительную диаграмму, подготовленную D.H.Brown Associates и Gartner Group, и отражающую функциональные возможности систем PDM различных производителей (рис. 2).

**Рисунок 2.**  
Функциональные возможности различных PDM систем.

## **Литература**

- [1]. *EDM Report, N 1, 2. 1995г.*
- [2]. *Interface, Vol.1, N. 3 1995*
- [3]. *PDM Byer"s Guide. CIMdata/Dressler Verlag GmbH. 1995.*
- [4]. *CIMdata PDM Europe"95 Conference. Thesis. Nordwijk, Netherlands 24 - 25 Oct. 1995*
- [5]. *Kalthoff Group EDMS Conference. Proceedings Chicago, IL, USA. 25 - 28 Sept. 1995*

## **6. Архивирование в Trace Mode 5**

### **ОТЧЕТ ТРЕВОГ**

Отчет тревог служит для записи в ASCII-файл информации об изменении значений атрибутов каналов, сообщения, содержащие тексты из словаря событий, и интерактивные сообщения оператора.

#### ***Механизм записи сообщений***

Сохранение сообщений в отчет тревог реализовано в виде отдельного потока с более низким приоритетом, чем пересчет базы каналов. МРВ формирует очередь сообщений для записи. Поток архивирования берет данные из этой очереди и записывает их на диск.

Если интенсивность потока сообщений превышает скорость их записи на диск, то очередь растет. По умолчанию предельный размер очереди равен 64000 сообщений. При достижении этого размера новые сообщения затирают самые старые.

Если очередь сообщений пуста, то файл отчета тревог закрывается без записи сообщений. При этом только обновляется FAT. При наличии сообщений в очереди файл снова открывается.

Отчет тревог может иметь размер до 4 Гбайт. По умолчанию его максимальный размер принимается равным 140 Мбайт. При достижении этого размера новые сообщения начинают записываться со второй строки. Для управления размером файла и длиной очереди используются системные каналы (см. ниже).

### ***Направления передачи сообщений***

Формируемые сообщения могут передаваться на ряд направлений. Существуют следующие направления посылки сообщений:

В файл отчета тревог – направление **AR**;

В графические консоли – направление **G**;

Программируется – направление **PRN**;

Программируется – направление **M**.

Для настройки направлений **PRN** и **M** в директории проекта надо создать файл **alasead.cfg**. В его первых двух строках описываются используемые направления:

**PRN** – на принтер, используемый по умолчанию (**PRN ANSI** для распечатки в кодировке ANSI);

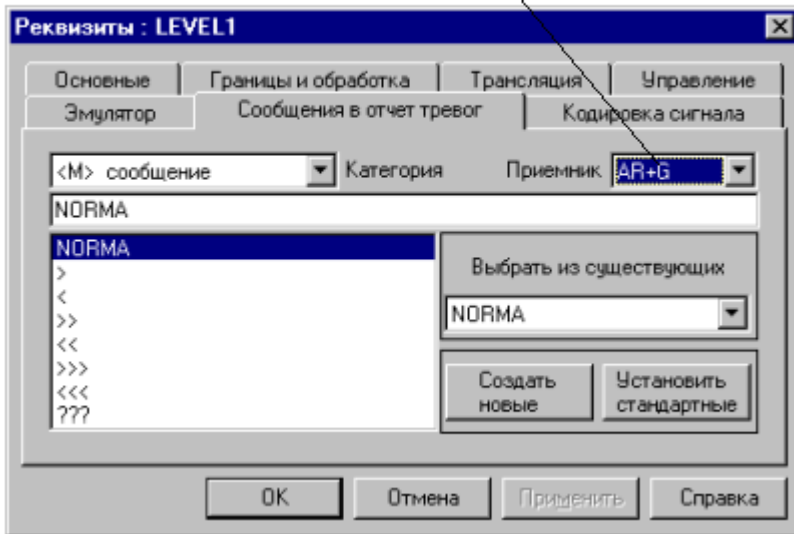
**GSM** – SMS сообщение на сотовый телефон;

**<имя\_файла> ANSI** – в указанный файл в кодировке ANSI.

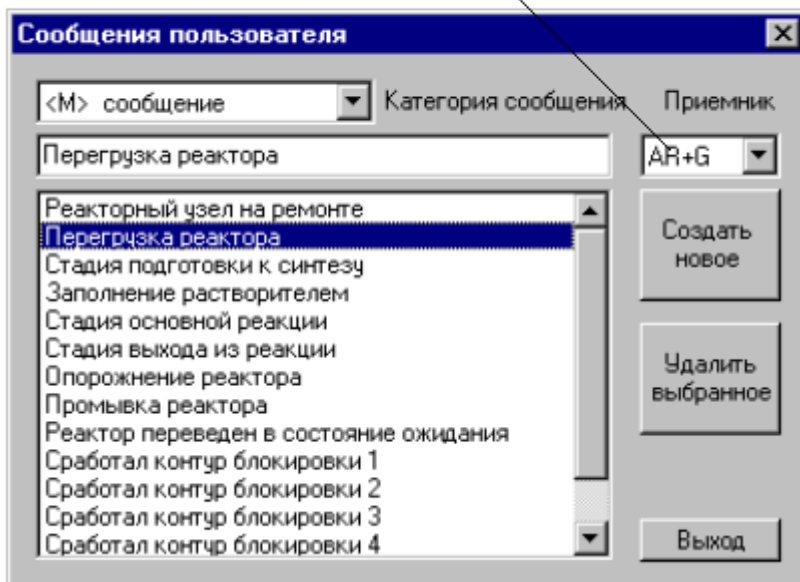
Файл **alasead.cfg** должен завершаться пустой строкой.

Выбор направлений передачи сообщения осуществляется в бланке **Сообщения в отчет тревог** диалога **Реквизиты** и диалоге **Сообщения пользователя**, показанных на следующих рисунках.

Выбор направления  
передачи сообщений



Выбор направления  
передачи сообщений



### *Формат строки отчета тревог*

Сообщения, заносимые в отчет тревог, оформляются в виде строк фиксированной длины - 136 символов. Каждая строка состоит из набора полей, разделенных пробелами:

Дата Время ИД Имя Код Сообщение Икв Ткв Номер

где

<b>Дата</b>	дата формирования строки [ДД-ММ-ГГ]: <b>ДД</b> - день месяца; <b>ММ</b> - месяц; <b>ГГГГ</b> - год;
<b>Время</b>	время формирования строки [ЧЧ:ММ:СС.Х];



	<b>ЧЧ</b> - часы; <b>ММ</b> - минуты; <b>СС</b> - секунды; <b>Х</b> - доли секунды;
<b>ИД</b>	символ идентификатора типа сообщения: <b>A</b> – аварийное сообщение; <b>W</b> – предупредительное сообщение; ...
<b>Имя</b>	имя канала (13 символов);
<b>Код</b>	кодировка канала или комментарий (21 символ);
<b>Сообщение</b>	текст сообщения (48 символов);
<b>Икв</b>	числовой идентификатор пользователя, квитирувавшего сообщение (4 символа);
<b>Ткв</b>	при квитировании сообщения в это поле заносится время в следующем формате [ <b>ДД_ЧЧ:ММ:СС</b> ]:  <b>ДД</b> - день месяца; <b>ЧЧ</b> - часы; <b>ММ</b> - минуты; <b>СС</b> - секунды;
<b>Номер</b>	индивидуальный номер строки в шестнадцатеричном виде (8 символов).

Формат первой строки отличается от остальных. В ней присутствуют дата и время создания отчета тревог, сообщение **START** и три числа, содержащих служебную информацию.

#### Квитирование строк отчета тревог

Поля **Икв** и **Ткв** строки отчета тревог предназначены для квитирования – подтверждения того, что оператор видел данное сообщение. В них заносится числовой идентификатор пользователя, квитирувавшего сообщение, дата и время квитиования.

Сообщения могут квиться в МРВ с помощью форм отображения, имеющих функции управления, с помощью формы просмотра ОТ, из специального окна просмотра ОТ, а так же из консоли тревог. При квитировании заполняются соответствующие поля строки отчета тревог.

#### Типы сообщений

Все сообщения в отчет тревог имеют тип. Он определяет степень ответственности сообщения и задается при вводе текста сообщения. При просмотре ОТ сообщения можно фильтровать по типу.

Определены следующие типы сообщений:

<**Пробел**> – без класса;

**M** – сообщение;

**W** – предупредительное сообщение;

**E** – ошибка;

**I** – информация;

**A** – аварийное сообщение;

**R** – изменение атрибутов канала;

**S** – пользовательское;

**Y** – пользовательское;

**0, ..., 9** – пользовательское;

**\_** – невидимое (не передается в графику);

**-** – неквитируемое;

**!** – командное;

**?** – резерв;

**\*** – системное невидимое.

**Внимание!** Сообщение, которое начинается со знака @, не выводится в отчет тревог. Это особенно существенно для битовых сообщений.

### ***Сообщения по реальным значениям***

У архивируемых в отчет тревог каналов контролируются изменения их реального значения. По результатам формируются сообщения, для записи в ОТ. В зависимости от вида представления канала режим формирования сообщений будет отличаться. Как известно, существует два вида представления: число с плавающей точкой (**F**) и целое 16-битовое число (**H**).

### **Контроль аналоговых параметров**

Аналоговые параметры обрабатываются каналами с видом представления **F**. Для них сообщения в отчет тревог заносятся при пересечении реальным значением аварийных границ и шкалы.

Для канала можно ввести величину гистерезиса на отслеживание границ. Если, например, значение канала пересекает верхнюю внутреннюю границу, то номер интервала меняется с 0 на 1 и формируется сообщение. При обратном изменении значения канала сообщение формируется после того, как реальное значение станет меньше границы на величину гистерезиса.

Величины аварийных границ и гистерезиса задаются в бланке **Границы и обработка диалога Реквизиты**, показанного на следующем рисунке.



### Контроль дискретных параметров

Эти параметры контролируются каналами с видом представления **Н**. Здесь на каждое изменение значения любого его бита формируется свое сообщение. Количество формируемых сообщений определяется числом битов, изменивших свое значение.

Для каждого бита реального значения каналов с видом представления **Н** определены два сообщения. Одно из них заносится в отчет тревог при изменении значения бита с 0 на 1, а второе - с 1 на 0.

Число контролируемых битов задается в бланке **Маски и эмуляция** диалога **Реквизиты**, как показано на рисунке.



### Формирование текстов сообщений по каналам

[Каналы с видом представления F](#)

[Каналы с видом представления Н](#)

[Формирование произвольных сообщений](#)

Текстовая строка поля **Сообщение** содержит описание возникшей ситуации. Эти строки задаются в бланке **Сообщения в отчет тревог** диалога **Реквизиты**, показанном на следующем рисунке



Тексты сообщений выбираются из системного словаря. Он содержит 40 стандартных сообщений. Первые 8 – для каналов с видом представления F, остальные – для H. Сообщения в списке бланка их настройки располагаются в строго определенном порядке. Вместо стандартных для каждого канала можно ввести собственные сообщения. Они сохраняются в системном словаре.

## ГЛОБАЛЬНЫЙ АРХИВ РЕГИСТРАТОР

Этот архив является общим для всего проекта. В него могут по сети сохранять данные все узлы. Сохранение данных в регистраторе обеспечивает монитор глобального регистратора (ГР).

Регистратор предназначен для сохранения в бинарном виде информации об изменениях значений каналов. В нем фиксируются изменения реального значения и всех невычисляемых атрибутов: период, границы, маски и настройки первичной обработки, а также флаги достоверности, состояния и подключения. Точность фиксации времени составляет 0.001 секунды.

ГР имеет фиксированный групповой номер – **200**. Он принимает данные, посылаемые по сети на этот номер, и сохраняет их. Поэтому в рамках проекта может существовать только один регистратор. Однако он может быть дублированным. В этом случае оба ГР одновременно принимают данные, посылаемые для сохранения. Каждый из них ведет свой файл архива, но эти файлы идентичны.

**Внимание!** При создании узла **Глобальный регистратор** ему по умолчанию присваивается имя вида **NODEn**, где **n** – порядковый номер. Чтобы узел имел групповой номер **200**, это имя не следует изменять.

При работе в дублированном режиме может автоматически осуществляться синхронизация архивов.

### Настройка канала для сохранения

Чтобы информация об изменении значения канала сохранялась в глобальном архиве, для него надо установить флаг **Регистратор**. Он устанавливается в бланке **Основные** диалога **Реквизиты**.

### Настройка параметров сохранения

В бланке **Архивация** диалога **Параметры узла** задаются два параметра сохранения данных в регистратор. Это флаг поддержки данного архива и признак начального состояния архивирования.

Поддержка глобального архива включается установкой регистратора в состояние **Активен**. В этом случае в базе каналов ГР для данного узла создается объект с соответствующим именем. В нем создаются каналы, принимающие архивируемые данные. Имена этих каналов воспроизводят имена архивируемых каналов в МРВ.

Чтобы при запуске МРВ отключить сохранение данных в регистратор следует поставить признак начального состояния **Выключен**.

### ***Управление в МРВ посылкой данных в ГР***

Сохранением данных в глобальный регистратор можно управлять при работе в реальном времени. Для управления сохранением данных в архивы предусмотрен специальный канал. Этот канал имеет подтип **СИСТЕМНЫЙ** и дополнение к подтипу **Архивация**. Для управления сохранением в глобальный регистратор используется второй бит данного канала (начиная с 0). Его значение задает следующие состояния архивирования:

**0** - включить запись данных в регистратор;

**1** - выключить запись данных в регистратор.

### ***Прием данных в регистраторе***

Глобальный архив для глобального регистратора является локальным архивом. Поэтому параметры архивирования, каналы для сохранения, а также функции управления и контроля настраиваются для него так же, как в МРВ для архива СПАД.

Глобальный архив фиксирует изменения реальных значений и всех невычисляемых атрибутов каналов со всех узлов проекта.

**Внимание!** Локальный архив ГР является глобальным архивом для остальных узлов проекта.

### ***Каналы для приема данных***

ГР принимает по сети данные, посланные на сохранение в его архиве. Для этого в его базе каналов создаются специальные каналы.

В базе каналов ГР создаются объекты, имеющие имена узлов, сохраняющих данные в регистратор. В этих объектах создаются каналы для приема архивируемых данных. Их имена совпадают с именами архивируемых каналов в соответствующих узлах. Эти каналы имеют подтип **СВЯЗЬ** и дополнение **IN Logger**. Для них устанавливается флаг сохранения в СПАД.

**Внимание!** Если канал-приемник архивируемых данных в ГР находится в состоянии **Выключен**, то никакие данные по каналу-источнику в регистратор не заносятся.

**Внимание!** Если канал-приемник включен, но переведен в состояние **Отключен**, то в регистратор не заносятся изменения реального значения канала-источника, а изменение его атрибутов фиксируется.

### ***Настройка параметров архива в ГР***

Параметры глобального архива настраиваются в редакторе базы каналов. Для этого используется диалог **Параметры узла** узла глобального регистратора. В этом диалоге для настройки архива используется раздел **СПАД** бланка **Архивация**. Здесь, как и для локального архива, следует указать имя файла архива, путь к этому файлу и его размер в мегабайтах.

### ***Управление архивированием в ГР***

Архивированием управляет канал **СИСТЕМНЫЙ** с дополнением **Архивация**. Значение его 0-го бита управляет разрешением записи в регистратор, а 8-го разрешением открытия файла архива:

**0** – разрешить;

**1** – запретить.

Запрет открытия файла используется при записи архива на сменный носитель во время его замены. При этом файл закрывается, а новые данные, накапливаются в буфере. После замены носителя значение 8-го бита следует снова установить равным 0. В результате на новом носителе создается файл архива. В него сохраняются данные из буфера и процесс архивирования продолжится.

**Внимание!** Если в канале-источнике одновременно установлены флажки **АВТОПОСЫЛКА** и **РЕГИСТРАТОР**, ГР использует сигналы автопосылок. Поэтому управлять сохранением таких каналов в архив ГР можно только в самом ГР или отменять в реальном времени сами автопосылки.

### ***Экспорт данных из регистратора***

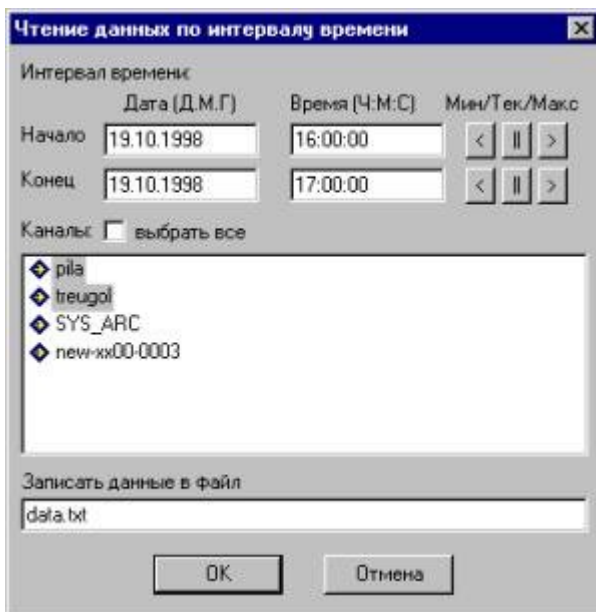
Данные из регистратора могут экспортироваться либо автоматически, с использованием специальных системных каналов в базе ГР, либо вручную по командам меню консоли управления ГР.

Автоматический экспорт из регистратора реализуется так же как из локальных архивов (СПАД) МРВ. Описание этих операций приведено выше в разделе **Экспорт данных из СПАД**.

### ***Реализация экспорта данных в ручном режиме***

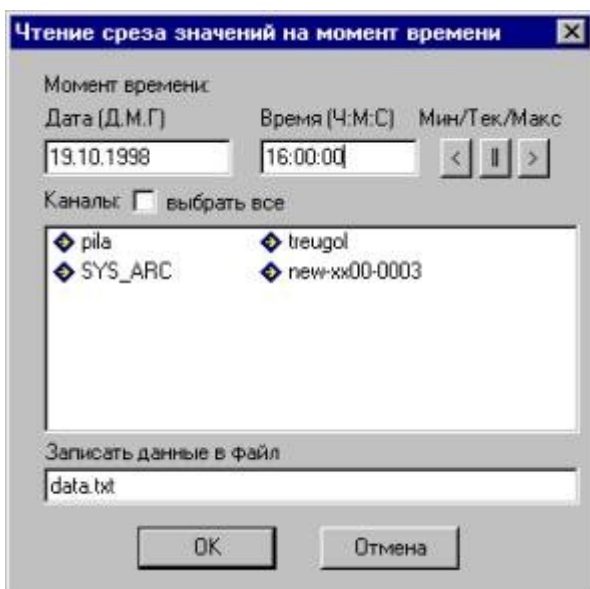
Существует два варианта экспорта данных: по временному срезу и по диапазону. Для реализации любого из них надо выполнить соответствующую команду в разделе **Экспорт данных** из меню **Управление** консоли управления ГР.

При выполнении команды **По интервалу** на экран выводится диалог, показанный на следующем рисунке.



Здесь следует указать временной интервал, каналы и имя файла для экспорта. По этим каналам в заданный файл будут записаны все изменения их значений за заданный диапазон времени.

При выполнении команды **По срезу** на экран выводится следующий диалог.



Здесь следует указать дату и время, каналы и имя файла для экспорта. По указанным каналам в заданный файл будут записаны их значения на заданное время.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 9. [Архивирование Trace Mode.](#)**  
**Справочная система Trace Mode 5, раздел: «Архивирование».**

# ЛЕКЦИЯ 10

## ОРГАНИЗАЦИЯ ВЗАИМОДЕЙСТВИЯ С ПРОМЫШЛЕННЫМИ КОНТРОЛЛЕРАМИ

1. Организация взаимодействия с промышленными контроллерами
2. Аппаратная реализация связи с устройствами ввода-вывода
3. Особенности построения коммуникационного ПО
4. Реальное время в SCADA – системах
5. Связь с УСО, контроллерами и приложениями в Trace Mode

- общие сведения;
- связь с УСО;
- связь с контроллерами;
- связь с приложениями (обмен DDE).

### 1. Организация взаимодействия с промышленными контроллерами

Современные SCADA-системы не ограничивают выбора аппаратуры нижнего уровня, так как предоставляют большой набор драйверов или серверов ввода-вывода и имеют хорошо развитые средства создания собственных программных модулей или драйверов новых устройств нижнего уровня. Сами драйверы разрабатываются с использованием стандартных языков программирования. Вопрос, однако, в том, достаточно ли только спецификаций доступа к ядру системы, поставляемых фирмой-разработчиком в штатном комплекте (система Trace Mode), или для создания драйверов необходимы специальные пакеты (системы FactoryLink, InTouch), или же, вообще, разработку драйвера нужно заказывать у фирмы-разработчика.

Для подсоединения драйверов ввода-вывода к SCADA используются два механизма - стандартный DDE (Dynamic Data Exchange - динамический обмен данными (технология для обмена данными в процессе работы приложений - низкоуровневое межзадачное взаимодействие в Windows)) и обмен по внутреннему (известному только фирме разработчику) протоколу. До сих пор DDE остается основным механизмом, используемым для связи с внешним миром в SCADA-системах. Но он является не совсем пригодным для обмена информацией в реальном масштабе времени из-за своих ограничений по производительности и надежности. Взамен DDE компания Microsoft предложила более эффективное и надежное средство передачи данных между процессами - OLE (Object Linking and Embedding - включение и встраивание объектов). Механизм OLE поддерживается в RSView, Fix, InTouch, Factory Link и др. На базе OLE появляется новый стандарт OPC (OLE for Process Control), ориентированный на рынок промышленной автоматизации. Новый стандарт, во-первых, позволяет объединять на уровне объектов различные системы управления и контроля, функционирующие в распределенной среде; во-вторых, OPC устраняет необходимость использования различного нестандартного оборудования и соответствующих коммуникационных программных драйверов. С точки зрения SCADA-систем появление OPC-серверов означает разработку программных стандартов обмена с технологическими устройствами. Поскольку производители полностью разбираются в своих устройствах, то эти спецификации являются для них руководством к разработке соответствующих серверов. Так как эти программные драйверы уже появляются на рынке, разработчики SCADA-систем предлагают свои механизмы связи с OPC-драйверами. OPC интерфейс допускает различные варианты обмена: получение "сырых" данных с физических устройств, из распределенной системы



управления или из любого приложения. На рынке появились инструментальные пакеты для написания OPC-компонентов, например, OPC-Toolkits фирмы FactorySoft Inc., включающий OPC Server Toolkit, OPC Client Toolkit, примеры OPC-программ.

Итак, для подсоединения драйверов ввода/вывода к SCADA - системе в настоящее время используются следующие механизмы:

- ставший стандартом de facto динамический обмен данными (DDE);
- собственные протоколы фирм-производителей SCADA - систем, реально обеспечивающие самый скоростной обмен данными;
- новый OPC - протокол, который, с одной стороны, является стандартным и поддерживается большинством SCADA - систем, а с другой стороны, лишен недостатков протоколов DDE.

DDE (Dynamic Data Exchange - динамический обмен данными) представляет собой коммуникационный протокол, разработанный компанией Microsoft для обмена данными между различными Windows - приложениями. Этот протокол реализует взаимосвязи типа клиент - сервер между двумя одновременно исполняющимися программами.

Изначально протокол DDE применялся в первых человеко - машинных интерфейсах (ММИ) в качестве механизма разделения данных между прикладными системами и устройствами типа ПЛК (программируемые логические контроллеры). Для преодоления недостатков DDE, прежде всего для повышения надежности и скорости обмена, разработчики предложили свои собственные решения (протоколы), такие как AdvancedDDE или FastDDE - протоколы, связанные с пакетированием информации при обмене с ПЛК и сетевыми контроллерами. Но такие частные решения приводят к ряду проблем:

- для каждой SCADA - системы пишется свой драйвер для поставляемого на рынок оборудования;
- в общем случае, два пакета не могут иметь доступ к одному драйверу в одно и то же время, поскольку каждый из них поддерживает обмен именно со своим драйвером.

Основная цель OPC стандарта (OLE for Process Control) заключается в определении механизма доступа к данным с любого устройства из приложений. OPC позволяет производителям оборудования поставлять программные компоненты, которые стандартным способом обеспечат клиентов данными с ПЛК. При широком распространении OPC - стандарта появляются следующие преимущества:

- OPC позволят определять на уровне объектов различные системы управления и контроля, работающие в распределенной гетерогенной среде;
- OPC - устранят необходимость использования различного нестандартного оборудования и соответствующих коммуникационных программных драйверов;
- у потребителя появится больший выбор при разработке приложений.

С OPC - решениями интеграция в гетерогенные (неоднородные) системы становится достаточно простой. Применительно к SCADA-системам OPC серверы, расположенные на всех компьютерах системы управления производственного предприятия, стандартным способом могут поставлять данные в программу визуализации, базы данных и т. п., уничтожая, в некотором смысле, само понятие неоднородной системы.

## **2. Аппаратная реализация связи с устройствами ввода-вывода.**

Для организации взаимодействия с контроллерами могут быть использованы следующие аппаратные средства:

- COM - порты.  
В этом случае контроллер или объединенные сетью контроллеры подключаются по протоколам RS-232, RS-422, RS-485.
- Сетевые платы.  
Использование такой аппаратной поддержки возможно, если соответствующие контроллеры снабжены интерфейсным выходом на Ethernet.
- Вставные платы.  
В этом случае протокол взаимодействия определяется платой и может быть уникальным. В настоящее время предлагаются реализации в стандартах ISA, PCI, CompactPCI.

### 3. Особенности построения коммуникационного ПО.

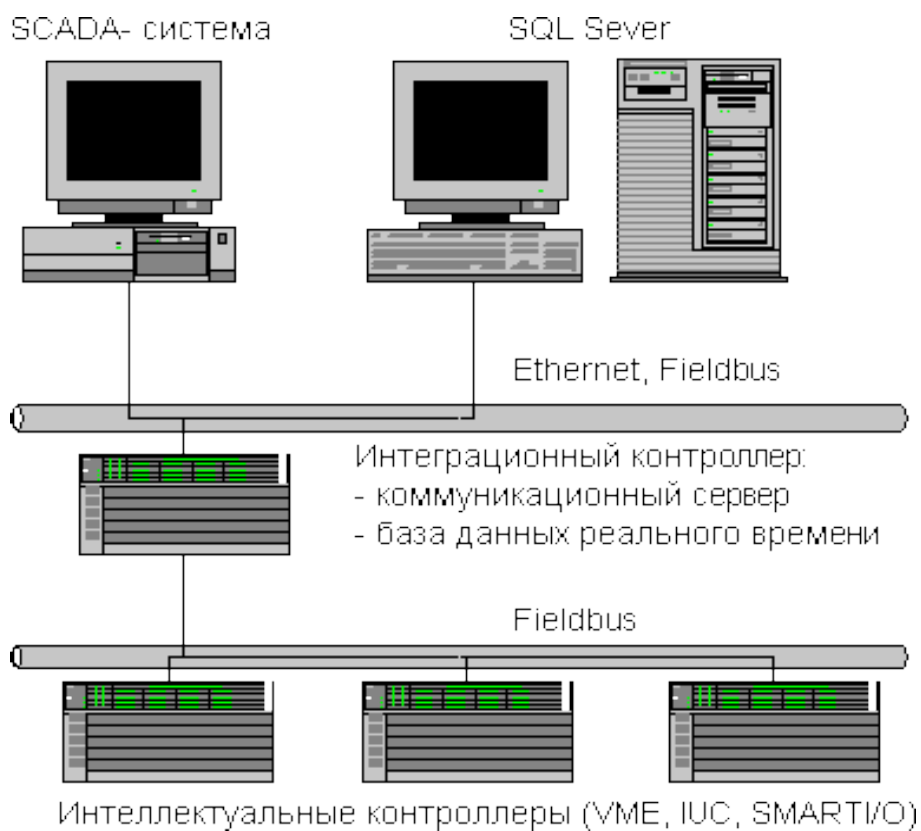


Рис.5. Типовая архитектура системы управления.

Коммуникационное программное обеспечение является многоуровневым. Количество уровней зависит от используемой операционной системы. Так, Applicom предлагает поддержку для следующих ОС: MS-DOS, UNIX SCO, HP-UX V10, OS/2, MS Windows 3.x, Windows 95/98, Windows NT4 на Intel и Alpha-платформах. Для Windows-платформ ПО включает следующие типы:

- статическая библиотека, используемая с традиционными языками программирования, такими как C, C++, Pascal;
- DLL (динамическая библиотека), применяемая со всеми Windows языками программирования (Visual Basic, Visual C/C++, Borland C/C++, Delphi, LabWindows CVI, LabView);
- DDE-сервер (имеет 16 и 32 битные реализации);

- пакетные реализации DDE протокола - FastDDE для продуктов линии Wonderware и AdvancedDDE для Rockwell линии;
- SuiteLink сервер, реализующий механизм обмена по SuiteLink протоколу, используемому компонентами пакета FactorySuite (Wonderware);
- OPC-сервер, поддерживающий интерфейс, определенный OPC- спецификацией.

На рис.6 показаны программные интерфейсы для Windows-приложений (в том числе и SCADA-систем) и спектр широко распространенных промышленных протоколов. Использование этих протоколов позволяет организовать взаимодействие с контроллерами, устройствами, объединенными промышленными (fieldbuses) и обычными сетями. Предлагаемая схема решения позволяет конечному пользователю, системному интегратору, единообразным способом организовать взаимодействие между ПО верхнего уровня и платами, специфичными для каждого типа промышленных сетей.

DDE, OPC - компоненты являются серверами по отношению к SCADA - системам. По отношению к ПО нижнего уровня (fieldbus) возможна организация Master/Slave и Client/Server. Внешние устройства способны посылать и принимать данные через плату. Когда вставляемая в персональный компьютер плата является Master/Client, то именно плата с поддерживаемым ПО является инициатором опроса промышленных устройств. В случае применения плат типа Slave/Server они реагируют на запросы внешних устройств. На некоторых вставных платах имеется разделяемая область памяти. Эта память доступна как приложению в ПК, так и встраиваемому ПО.

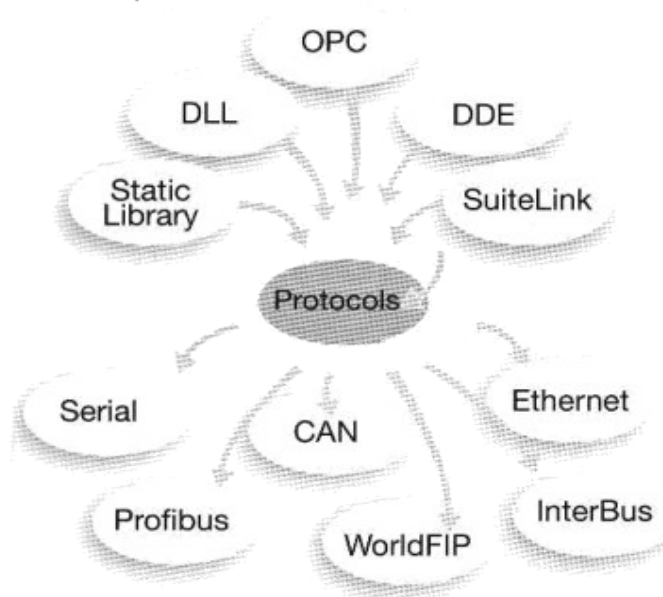


Рис.6. Набор интерфейсов для SCADA - систем и спектр поддерживаемых протоколов.

На рис.7 показана обобщенная схема организации коммуникационного ПО для Windows NT.

На предлагаемой схеме отражены как традиционные решения на базе стандартных Windows NT - драйверов, так и с использованием библиотек, реализованных в расширении реального времени RTX от VenturCom.

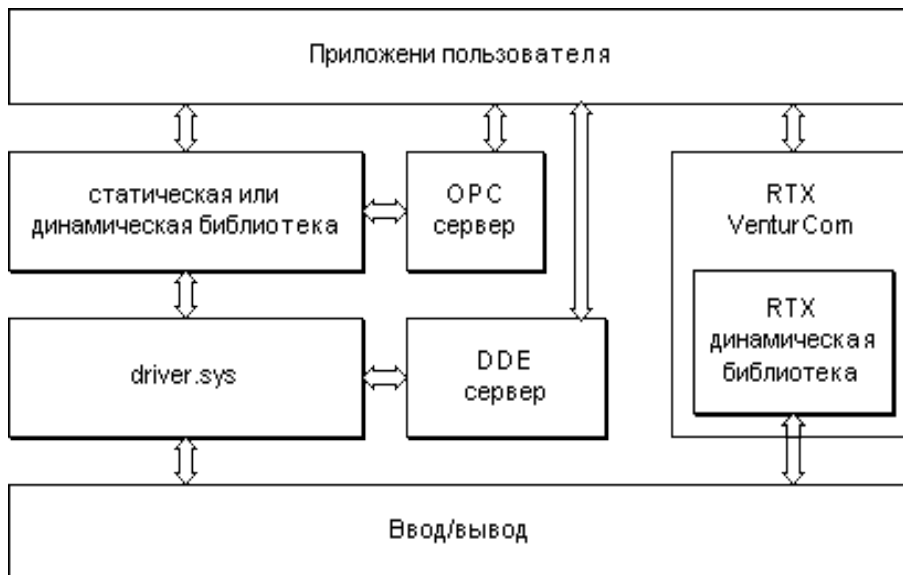


Рис.7. Схема организации коммуникационного ПО для Windows NT.

#### 4. Реальное время в SCADA – системах

Одним из существенных недостатков SCADA-систем на платформах Windows 9x по сравнению с таковыми же системами на платформах ОСРВ является отсутствие поддержки реального времени. Ситуация резко изменилась с появлением Windows NT. Выход в свет этой ОС стимулировал разработку новых подходов в поддержке жесткого реального времени. Прежде всего, сама по себе Windows NT весьма успешно теснит ОСРВ. Тем не менее, Windows NT имеет ряд ограничений. Такие ее особенности, как: предпочтение аппаратного прерывания над программным (даже если это простое движение мыши), выполнение в подпрограмме обработки аппаратных прерываний лишь необходимых действий с выполнением последующей обработки через очередь отложенных процедур, отсутствие приоритетной обработки процессов в очереди отложенных процедур, не позволяют отнести Windows NT к категории классических ОС реального времени.

Ряд фирм (LP Elektronik, Imagination Systems, RadSys, Spectron Microsystems, VenturCom) предприняли более радикальные попытки превратить Windows NT в ОС жесткого реального времени. Рассмотрим некоторые ключевые особенности реализации такой идеи на подсистеме реального времени RTX (Real Time Extension), предложенной фирмой Ventur Com. Именно эта реализация получает в настоящее время наиболее широкое распространение. Фирмы-разработчики SCADA-систем незамедлительно начали предлагать применение новых решений. Так, набор прикладных интерфейсов программирования RTX 4.1 (Ventur Com) в FIX позволяет:

- Осуществлять полный контроль над задачами реального времени;
- Использовать фиксированную систему из 128 приоритетов для контроля RTX задач;
- Применять стандартные средства обмена данными между задачами;
- Обращаться к стандартным функциям из Win32 API.

Появление подобных решений наряду с собственными характеристиками Windows NT наносит сильный "удар" по SCADA-системам на базе ОСРВ, поскольку отнимает у них очень важный "козырь" - преимущества жесткого реального времени, и, для некоторых приложений, теснит применение ОСРВ во встраиваемых системах. Сегодня для весьма широкого спектра промышленных приложений уже есть реальная возможность

использовать WindowsNT как единую операционную систему, со всеми вытекающими отсюда последствиями, работающую даже в бездисковых конфигурациях в сетевых контроллерах ввода/вывода, скажем на платформе CompactPCI или VME, наряду с использованием на рабочем месте оператора-технолога.

#### 4. Связь с УСО, контроллерами и приложениями в Trace Mode 5

##### **Общие сведения**

В составе системы ТРЕЙС МОУД имеются мониторы реального времени для операторских станций и PC-контроллеров. Для обмена данными между ними используются протоколы M\_LINK и I-NET. Связь мониторов ТРЕЙС МОУД по этим протоколам описана в разделе **Разработка распределенных систем**.

MPB поддерживает обмен данными с большим количеством контроллеров, использующих собственные средства программирования и имеющих свои протоколы обмена. Часть этих протоколов встроена в исполнительный модуль, а часть поставляется опционально в виде драйверов.

Микро MPB поддерживает только встроенные протоколы, однако набор поддерживаемых плат УСО в Микро MPB шире, чем в MPB.

В данном разделе будут рассмотрены следующие темы:

обмен данными с контроллерами и платами УСО по встроенным и внешним протоколам;

обмен данными с внешними приложениями с использованием механизмов DDE, OPC и ODBC.

##### **Связь с контроллерами**

ТРЕЙС МОУД поддерживает обмен данными с разными контроллерами. Для PC-контроллеров обмен реализуется по собственным протоколам ТРЕЙС МОУД при использовании в них Микро MPB, а для остальных – по их протоколам. Часть этих протоколов встроена в исполнительные модули ТРЕЙС МОУД, а часть поставляется опционально в виде динамически загружаемых библиотек.

##### ***Встроенные протоколы***

Для обмена по встроенным протоколам предусмотрены следующие каналы:

[подтипа СВЯЗЬ](#) ;

[подтипа DCS](#) ;

[подтипа MODBUS](#) .

Первый из них используется мониторами ТРЕЙС МОУД для обмена между собой. Его описание приведено в разделе **Разработка распределенных систем**. Рассмотрим два последних подтипа.

Микро МРВ поддерживает до 4 связей со статусом MASTER по M-Link или по другому встроенному протоколу (по 4 COM-портам, имеющим один и тот же вектор прерывания).

### Обмен с модулями распределенного УСО

Для связи с модулями распределенного УСО типа LAGOON, ROBO, ADAM-4000 и ADAM-5000/485, NuDAM-6000, I-7000, RIO-2000 и подобными используются каналы [подтипа DCS](#). Дополнение к подтипу этого канала определяет запрашиваемые или передаваемые данные.

**Внимание!** Микро МРВ не поддерживает обмен с контроллерами ADAM 5000.

### Обмен по протоколу MODBUS

Обмен данными с контроллерами, поддерживающими протокол MODBUS, реализуется с помощью каналов [подтипа MODBUS](#). При этом код команды в запросе определяется дополнением к подтипу этого канала.

В ТРЕЙС МОУД адрес переменной типа **Float** в протоколе MODBUS представляет собой номер первого из двух слов, составляющих эту переменную, в массиве слов. Массива переменных **Float** в MODBUS нет. Таким образом, если в контроллере сформирован массив переменных **Float**, начиная с 30-го слова, то их адреса в базе должны быть 30, 32, 34 и т.д.

4 байта переменной типа **Float** передаются в кадрах функций 3, 4 и 16 в следующем порядке:



### Обмен по протоколу MODBUS TCP/IP

Обмен данными с контроллерами, поддерживающими протокол MODBUS TCP/IP, реализуется с помощью каналов [подтипа MODBUS](#) с настройкой **TYPE=TCP**. Данное значение настройки устанавливается при загрузке автоматически, если не задано значение настройки **#RS**. Настройку **TYPE** можно также задать вручную.

Номер устройства и IP адрес задаются в файле с именем **IP\_modBus** (без расширения). Строка файла имеет следующий формат:

<id> <IP\_address>

где

<id> – номер устройства;

<IP\_address> – IP адрес в стандартном формате. IP адрес не должен содержать 0 в левой позиции – в этом случае происходит его преобразование в восьмеричное число.

Если описание устройства в файле отсутствует, каналу выставляется признак недоверности и он отключается.

Если ответ содержит информацию об ошибке, то канал подтипа **Диагностика** с дополнением **Modbus** принимает значение 7.

Ошибки протокола записываются в канал подтипа **Диагностика** с дополнением **Error IP**.

Код команды в запросе определяется дополнением к подтипу канала. Максимальная длина группового запроса – 255.

Для соединения используется порт 502 – стандартный порт для MODBUS TCP/IP.

### ***Внешние протоколы***

Обмен с использованием внешних протоколов реализован только в мониторе реального времени. Микро МРВ не поддерживает эту функцию.

Для обмена данными по внешним протоколам используются каналы подтипов **КОНТР\_1** и **КОНТР\_2**. Их дополнение к подтипу задает тип контроллера. Список его значений модифицируется по мере добавления в систему новых драйверов.

Каналы **подтипа КОНТР\_1** предназначены для связи по последовательному интерфейсу, а каналы **подтипа КОНТР\_2** используются при необходимости описать носитель протокола внешними средствами (например, собственная интерфейсная плата). Поэтому в первом случае требуется один драйвер, описывающий протокол, а во втором – два. Первый используется для описания протокола, а второй – носителя.

Руководство по разработке драйверов для обмена данными с контроллерами приведено в разделе **Разработка драйверов. Интерфейс ТСОМ**.

### **Обмен данными с контроллерами РЕМИКОНТ-130**

Обмен данными с контроллерами типа РЕМИКОНТ-130 реализуется с помощью каналов подтипа **КОНТР\_1** с дополнением к подтипу **РЕМИКОНТ-130**.

Эти каналы имеют следующие настройки:

**№порта** – номер последовательного порта, по которому осуществляется обмен с контроллером (0 – СОМ1, ..., 31 – СОМ32). Этот параметр задается в формате HEX;

**№контр-ра** – номер контроллера в сети "Транзит";

**Переменная** – тип запрашиваемой или управляемой переменной. Тип выбирается из списка, содержащего следующие пункты:

**NET** – опрос сетевых переменных. Для данного типа команды в настройке **№вх/вых** задается номер контролируемой переменной;

**КНФ** – опрос конфигурации сети (16-ти битовое целое число, каждый бит которого является идентификатором наличия в сети контроллера с соответствующим номером);

**ВХ** – вход алгоблока;

**ВЫХ** – выход алгоблока;

**КНТ** – опрос состояния контура. Для данного типа команды в настройке **№алгоблока** указывается номер контура, а величина настройки **№вх/вых** определяет контролируемую характеристику контура:

**1** – величина ручного задания;

**2** – текущее значение задания;

**3** – вход регулятора;

**4** – рассогласование;

**5** – выход регулятора;

**6** – режим работы контура:

**13,14 биты:**

00 – ЗДЛ отсутствует;

01 – локальный режим;

10 – каскадный режим;

**1 бит** – признак ошибки контура;

**2 бит** – наличие алгоритма РУЧ;

**3 бит:**

0 – локальный или каскадный;

1 – дистанционный режим;

**4 бит:**

0 – автоматический режим;

1 – ручной режим;

**5,6 биты:**

00 – ЗДН отсутствует;

01 – внешнее задание;

10 – программное задание;



11 – ручное задание;

**РУЧ** – управление значением ручного задания;

**У\_В** – управление выходным сигналом;

**РЕГ** – управление режимом контура. Режим задается значением канала:

**07** – установка режима ВЗ;

**09** – установка режима ПЗ;

**0В** – установка режима РЗ;

**0D** – переход на автоматический режим;

**0F** – установка режима РУ;

**11** – установка режима ДУ;

**13** – отмена дистанционного управления;

**15** – установка режима КУ;

**17** – установка режима ЛУ;

**№алгоблока** – номер алгоблока в пользовательской программе контроллера для переменных ВХ и ВЫХ; для переменных КНТ, РУЧ, У\_В и РЕГ – номер контура регулирования;

**№вх/вых** – номер запрашиваемой или управляемой переменной для переменных NET, КНТ, ВХ и ВЫХ. Номера индексов каналов;

**Описание** – вид представления переменной.

Для обмена необходимо настроить последовательный порт (см. раздел [Настройка последовательных портов](#)).

## **СВЯЗЬ С ПРИЛОЖЕНИЯМИ**

### **Обмен по DDE**

Одним из механизмов обмена данными приложений WINDOWS является DDE (Dynamic Data Exchange). МРВ может выступать как в качестве DDE-сервера, так и DDE-клиента и поддерживает три режима обмена: **ADVISE**, **POKE**, **REQUEST**.

#### ***Работа МРВ в качестве DDE-сервера***

Выступая DDE-сервером, МРВ выполняет следующие операции:

    посылает реальные значения каналов приложению по своей инициативе (режим **ADVISE**);

изменяет значения атрибутов каналов по команде другого приложения (режим **POKE**);

посылает значения любых атрибутов каналов по запросу другого приложения (режим **REQUEST**).

### Обмен данными на локальной машине

Чтобы обмениваться данными с MPB, используя механизм DDE, надо посылать ему запросы в следующем формате:

**<server>|<topic>!<item> <data>**

где:

**<server>** – имя DDE-сервера в формате **RTM<nnn>**, где **<nnn>** - номер узла, к которому адресован запрос;

**<topic>** – тема запроса. В режиме **ADVISE** указывается **GET**, а в режимах **POKE** и **REQUEST** – обозначения атрибутов каналов;

**<item>** – имя канала;

**<data>** – посылаемое значение в формате числа с плавающей запятой. Оно присутствует только в режиме **POKE**.

При запросе реального значения в режиме **REQUEST** в качестве темы надо указать **GET**.

Для отправки значения из приложения в атрибут ВХОД канала ТРЕЙС МОУД (режим **POKE**) надо задать тему **PUT** или **GET**.

Рассмотрим примеры использования DDE для связи MPB и Excel.

### Пример

Требуется запрашивать в режиме **ADVISE** значения канала TRCAS\_234 и помещать их в ячейку B5 таблицы Excel. MPB, у которого запрашиваются данные, имеет сетевой номер 5.

Для решения этой задачи в ячейку B5 надо занести следующий запрос:

**=RTM5|GET!TRCAS\_234**

### Пример

Требуется посылать из EXCEL в диалоге по нажатию кнопки в верхнюю границу шкалы значение 300.

Для решения данной задачи надо выполнить следующие действия:

- 1) занести в ячейку A1 таблицы "Лист1" значение 300;
- 2) написать короткий макрос на Visual Basic for Applications:

```

Sub MyDDE()

channelNumber = Application.DDEInitiate("RTM5", "HL")

Application.DDEPoke channelNumber, "TRCAS-234", _
Worksheets("Лист1").Cells(1,1)

Application.DDETerminate channelNumber

End Sub

```

3) создать диалог и обеспечить выполнение этого макроса при нажатии заданной кнопки в этом диалоге.

### **Обмен по NetDDE**

Для связи МРВ с другими приложениями, запущенными на удаленных компьютерах, можно использовать механизм NetDDE. Формат запроса в этом случае остается таким же, как описано выше. Однако элементы **server** и **topic** формируются иначе.

Имя сервера формируется следующим образом:

```
\\<NAME>\NDDES
```

где

<NAME> - имя компьютера, где работает МРВ.

В качестве темы надо записать **RTM<nnn>\$**, где <nnn> - номер узла, к которому адресован запрос, а в качестве **item** - имя канала.

**Внимание!** При обмене по NetDDE запрашиваются только реальные значения каналов и формируются только входные значения.

### **Контроль и управление обменом по DDE**

Управление обменом по DDE осуществляется каналами подтипа **СИСТЕМНЫЙ** с дополнением **Сеть, DDE**. Для этого используются биты с 6-го по 11-й (считая с 0).

Контроль текущего состояния обмена по DDE осуществляется с помощью канала **ДИАГНОСТИКА** с дополнением **DDE**. Эти каналы подробно описаны в приложении.

### **Работа МРВ в качестве DDE-клиента**

Выступая DDE-клиентом, МРВ выполняет следующие операции:

- принимает данные, посылаемые DDE-серверами, по мере их изменения (режим **ADVISE**);

- управляет переменными в DDE-серверах (режим **POKE**);

запрашивает данные у DDE-серверов (режим **REQUEST**).

### Организация обмена с DDE-серверами

Для обмена данными с DDE-серверами предусмотрен подтип каналов **DDE**. Его дополнение к подтипу указывает файл конфигурации обмена. Такие файлы должны располагаться в директории проекта. Их имена формируются следующим образом:

**DDECNF<n>.CNF**

где

**<n>** – номер от 0 до 7. Он соответствует значению дополнения к подтипу (если дополнение к подтипу DDE0, то n должно быть равно 0, если DDE1, то 1 и т. д.).

Настройки каналов подтипа DDE определяют режим обмена, а также метод и режим формирования запросов у DDE-сервера.

### Формат файла конфигурации DDE-обмена

Этот файл имеет текстовый формат и содержит следующую информацию:

server topic0 [format\_item]

topic1

...

где

**server** – имя DDE-сервера;

**topic<n>** – название темы (до 256 тем, которые выбираются настройкой канала D (см. ниже));

**format\_item** – формат элемента **item**.

Формат **item** может отсутствовать. В этом случае для канала надо выбрать режим использования в качестве **item** его имени.

Строка описания элемента **item** может содержать тексты и форматы вывода чисел в нотации языка Си. При формировании DDE запроса **item** будет состоять из текстов и вставленных в него чисел в указанном формате. Величины чисел определяются значением настроек канала. Например, в запросе данных от OLE/DDE менеджера фирмы SIEMENS при связи по PROFIBUS DP формат элемента **item** выглядит следующим образом:

Slave<aaa>EB<bbb>

где

**<aaa>** - три цифры определяющие номер устройства в сети PROFIBUS;

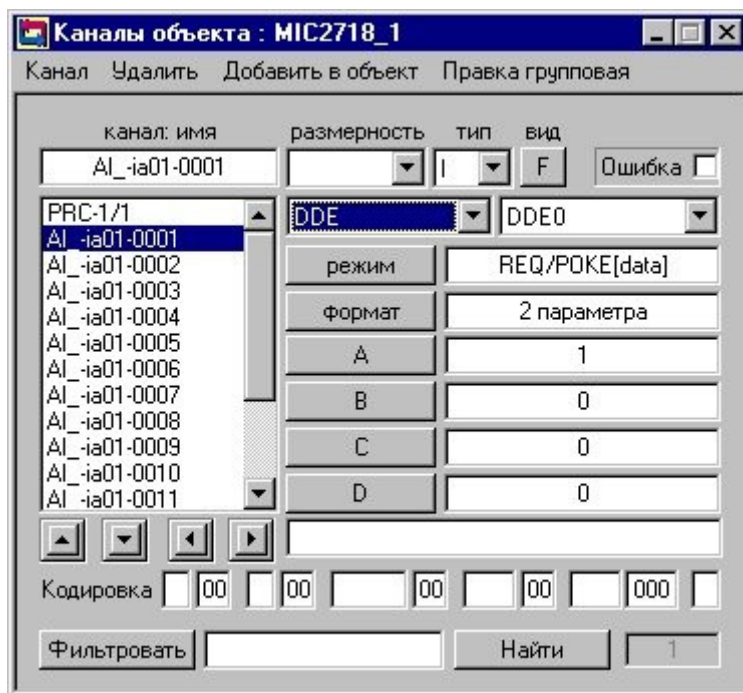
<bbb>- три цифры определяющие номер запрашиваемой переменной на указанном устройстве.

Формат элемента **item** при этом выглядит следующим образом:

Slave%.3dEB%.3d

### Настройки каналов подтипа DDE

Каналы подтипа DDE имеют шесть настроек. Следующий рисунок демонстрирует настройку каналов данного подтипа.



Первая настройка называется **режим**. Она используется для задания режима обмена. Он выбирается из списка, содержащего следующие пункты:

**REQ/POKE[data]** – обмен в режиме REQUEST для каналов типа INPUT и POKE для каналов типа OUTPUT;

**REQ/POKE[data/r]** – обмен в режиме REQUEST для каналов типа INPUT и POKE для каналов типа OUTPUT с добавлением в конце передачи данных символа /r;

**REQ/POKE[data/n]** – обмен в режиме REQUEST для каналов типа INPUT и POKE для каналов типа OUTPUT с добавлением в конце передачи данных символа /n;

**ADVISE** – обмен в режиме ADVISE (только для каналов типа INPUT).

Следующая настройка (**формат**) определяет метод формирования элемента **item** в запросе. Метод выбирается из списка, содержащего следующие пункты:

**Имя** – в качестве **item** используется имя канала;

**2 параметра** – в строке описания формата элемента **item** может присутствовать до двух полей вставки чисел. Значение первого числа определяется величинами настроек А и В (младший и старший байт соответственно), а второго – С;

**3 параметра** – в строке описания формата элемента **item** может присутствовать до трех полей вставки чисел. Значения первого, второго и третьего чисел определяются величинами настроек А, В и С соответственно;

**4 параметра** – в строке описания формата элемента **item** может присутствовать до четырех полей вставки чисел. Значения этих чисел определяются величинами настроек А, В, С и D.

**Внимание!** Каждая из настроек А, В, С и D являются целыми числами длиной в один байт. Соответственно их значения могут лежать в диапазоне от 0 до 255.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 10.** [Связь с УСО, контроллерами и приложениями Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Связь с УСО, контроллерами и приложениями».

## ЛЕКЦИЯ 11 СИСТЕМЫ PLM

1. Назначение PLM систем.
2. Производители PLM систем.
3. Потребители PLM систем.
4. Работа в реальном времени в Trace Mode 5
  - запуск MPB;
  - запуск графической консоли;
  - запуск сервера матобработки;
  - настройка DCOM;
  - структура обработки данных.

### Автоматизация: от идеи до утилизации

Наталья Дубова  
27.06.2003

[Открытые системы, #06/2003](#)

**По данным аналитиков, рынок PLM-решений ожидает ежегодный рост на 11%, — несмотря на проблемы, которые продолжает испытывать мировая экономика, стабильное увеличение числа инсталляций систем PLM не вызывает сомнений. В основе развития этого рынка лежат новые задачи, которые необходимо решать производственным компаниям в условиях усиливающейся конкуренции.**

В основе высоких темпов развития рынка PLM лежат новые задачи, стоящие перед производственными компаниями в условиях усиливающейся конкуренции. Попробуем, не вдаваясь в детали отдельных реализаций, но с точки зрения общих идейных основ и тенденций этого рынка, ответить на вопрос, в чем суть этих задач, и как их помогают решить системы класса PLM.

### Что есть PLM

Главная цель любого производителя — качество выпускаемой продукции, однако одной из основных тенденций сегодня является необходимость сокращения времени выхода новых изделий на рынок при одновременном удовлетворении специфических потребностей заказчиков (от модели массового производства компании переходят к разработке и сборке на заказ). При этом компании не могут позволить себе повышать стоимость процессов проектирования новых изделий. Одновременно с этим в процесс создания продукции вовлекаются сегодня многочисленные внешние участники — от поставщиков комплектующих, которые должны иметь возможность оперативно реагировать на изменения в требованиях к конечному продукту, до самих заказчиков, которые хотят получить доступ к процессам формирования этих требований. Для многих крупных производителей «виртуальное предприятие» становится реальностью — они выносят за скобки собственного производственного процесса разработку и выпуск комплектующих, а подчас и собственно сборку готового изделия, оставляя за собой базовые операции выработки концепции и проектирования продукции. Передача части своих функций на аутсорсинг не отменяет необходимости контролировать и

интегрировать все процессы. Для того чтобы виртуализация производства происходила не в ущерб конечному результату и с максимальной экономической отдачей, компаниям необходимы технологии, объединяющие и автоматизирующие все разрозненные этапы жизненного цикла изделия, создающие интегрированную среду коллективной работы, где каждый участник производственной цепочки имеет в реальном времени доступ к нужной ему информации по изделию.

Первоначально такие технологии приобрели известность под названием Collaborative Product Commerce (CPC), однако термин PLM (Product Lifecycle Management) точнее отражает суть. PLM — это набор взаимосвязанных прикладных решений, включающий необходимые программные компоненты обеспечения коммуникаций, интеграции модулей, автоматизированного проектирования и визуализации и других решений, охватывающих полный жизненный цикл продукта — от идеи до утилизации. PLM расширяет возможности автоматизированного контроля над изделием за рамки инженерных лабораторий и конструкторских бюро, которые были основными пользователями предшественников PLM-технологий — CAD/CAM и систем PDM. Решения класса PLM призваны объединить всех участников жизненного цикла, как внутри предприятия-производителя, так и вне его, в том числе поставщиков, заказчиков и организации, занятые послепродажным обслуживанием продукции.

Система PLM охватывает все этапы жизненного цикла изделия (рис. 1).



Рис. 1. Этапы жизненного цикла изделия

**Выработка концепции проекта.** На основе анализа требований рынка формируется общая идея нового изделия или, что случается значительно чаще, концепция усовершенствований в проекте уже существующего продукта. Система PLM предоставляет информацию, которая может использоваться для анализа жизнеспособности полученной концепции.

**Анализ требований рынка.** Производитель должен понять, насколько востребован рынком новый продукт, и оценить выполнимость этих требований. На этом этапе система PLM используется для извлечения данных из различных информационных систем, которые могут способствовать получению более точной картины.

**Проектирование.** Конструкторы создают проект нового изделия — соответствующие САПР и PDM-решения являются интегральной частью PLM-решения. При проектировании используется вся необходимая дополнительная информация, поставщиком которой являются PLM-модули, включая факторы, связанные с послепродажным обслуживанием изделия, информация о предпочтениях заказчика, данные о производственных возможностях и т.д.

**Определение источников поставок (PLM-sourcing).** Отдел закупок должен провести предварительную работу по поиску источников приобретения необходимых для производства изделия деталей, материалов, компонентов, оборудования и т.д. Задача



систем PLM — предоставить достоверные данные о доступности тех или иных деталей/компонентов/материалов, их стоимости, потенциальных поставщиках и возможных альтернативных источниках.

**Производство.** В соответствии с определенными на этапе проектирования спецификациями и с использованием полученных на этапе поставок деталей и материалов производится продукт. Реализованные в PLM специальные методы контроля качества позволяют гарантировать соответствие производимого изделия заданным спецификациям.

**Дистрибуция.** Готовое изделие поставляется либо дистрибутору, который размещает его на своем складе до поступления соответствующего заказа, либо непосредственно заказчику. Полученные из системы PLM исторические данные о потребностях рынка помогают производителю свести к минимуму число уровней инвентаризации готовой продукции.

**Послепродажное обслуживание.** На этом этапе выполняются техническое сопровождение, обслуживание и ремонт — в течение гарантийного срока или как дополнительно оплачиваемый сервис. PLM позволяет учесть различную информацию об изделии, поступающую на этом этапе жизненного цикла, при разработке последующих проектов и тем самым способствует повышению привлекательности продукции для клиентов.

Возможность извлечь из одного источника не только все накопленные на данном проекте знания, но и исторические данные, имеет ключевое значение для повышения эффективности проектирования новой продукции и усовершенствования существующей номенклатуры изделий с целью максимального удовлетворения потребностей заказчиков. Репозиторий PLM позволяет производителю не растерять ценный опыт, накопленный на предыдущих проектах. Возможность не изобретать в очередной раз велосипед крайне важна для производственных предприятий, поскольку, как показывает практика, большинство проектов не несут в себе какой-то исключительной новизны, а являются усовершенствованиями предыдущих разработок.

При наличии централизованного репозитория значительно упрощается контроль за актуальностью информации — PLM становится для компании единственным источником достоверных данных по продукту. Чем раньше будут идентифицированы ошибки или ограничения проекта, которые могут повлиять на конечные характеристики разрабатываемого изделия или просто усложнить процесс его производства, тем меньше затрат понадобится на их устранение. Перепроектирование, а тем более, дополнительное тестирование, переделка или, того хуже, полная отбраковка готовой продукции всегда обходятся изготовителю недешево. Решения категории PLM позволяют свести к минимуму или полностью избежать подобных расходов, а если учесть, что, по оценкам Aberdeen, не менее 70% затрат на производство и сопровождение продукции приходится на этап проектирования, можно сделать вывод, что PLM обеспечивают не только повышение качества и оптимизацию разработки изделия, но и способствуют снижению затрат на поддержку его жизненного цикла. По данным IBM в сотрудничестве с компанией Dassault Systemes, предлагающей свой PLM-пакет ENOVIA, с применением возможностей PLM экономия затрат на разработку и выпуск продукции у ее клиентов достигает 1 млрд. долл., при этом цикл вывода нового изделия на рынок сокращается с 72 до 16 недель.

В PLM доступ к данным организован на ролевой основе. Система позволяет предоставлять пользователю информацию в форме, соответствующей выполняемым им

функциям в жизненном цикле изделия: трехмерные модели, схематические диаграммы, инженерные спецификации (bill of materials, BOM), календарные планы или прогнозы на основе анализа требований рынка. Конструктор будет работать в привычной ему среде САПР, а сотрудник маркетингового подразделения сможет получить из системы представление трехмерной сборки, пригодное для размещения в рекламной брошюре.

Системы CAD/CAM, нашедшие широкое применение в производственных компаниях, повышали эффективность и упрощали работу проектировщиков и технологов, но были слабо интегрированы между собой, а другие подразделения и партнеры компании практически не могли повлиять на процессы разработки и производства изделия, поскольку не имели доступа к таким системам. После последовательного проведения нескольких итераций разработки проект попадал к технологам и в отдел закупок, которые уже никак не могли повлиять на него и подчас были вынуждены идти на неоправданно дорогие траты или поиск дефицитных материалов и деталей и производство изделия на неподходящем оборудовании. Система PLM, объединяющая все структурные подразделения предприятия, позволяет уже на этапе проектирования вносить изменения с учетом проблем, которые могут возникнуть как при производстве, так и при выборе поставщиков комплектующих. В результате не возникнет ситуации, когда невозможность увязать проект со сложностями производственного цикла и задачами закупок приводит к повышению цены и одновременно снижению качества продукции.

С помощью информации, которую интегрирует система PLM, даже не обладая специальными техническими знаниями, сотрудники отдела закупок смогут заниматься поиском нужных деталей и выбором оптимальных каналов поставки непосредственно по данным, поступающим из конструкторских подразделений. По мере развития технологии электронного представления компонентов (component supplier management, CMS) и появления у поставщиков возможностей параметрического поиска, которые будут интегрироваться в PLM, сами конструкторы уже на этапе проектирования получат возможность выбирать подходящие компоненты. По данным Aberdeen, привлечение поставщиков на этапах формирования концепции и проектирования изделия обеспечивает 20% экономии средств на разработку. Кроме того, они все чаще выступают в роли проектировщиков определенных компонентов изделия. Бумажные формы обмена информацией с внешними контрагентами компании-производителя существенно снижают эффективность работы над изделием, поскольку время обнаружения ошибок в проекте влияет на его стоимость и на сроки вывода качественного продукта на рынок. Предположим, производитель автомобилей отдает на аутсорсинг разработку переднего амортизатора. В ходе проектирования корпуса может измениться спецификация для амортизатора — длина, вес или степень упругости. Эти изменения сразу должны быть учтены в проекте, который разрабатывает смежник, что и позволяет сделать среда PLM через механизмы разделения данных для разных участников жизненного цикла изделия.

Если идея интеграции поставщиков комплектующих в процессы проектирования на самых ранних стадиях для ряда производителей, например, в автомобильной отрасли, уже давно перешла в практику, то возможность включения в информационное пространство изделия подразделения и внешние организации, которые занимаются послепродажным обслуживанием, можно считать «ноу-хау» PLM-решений. Знания о том, какие проблемы вызывает техническое сопровождение готовой продукции, ее гарантийное или платное обслуживание, может серьезно повлиять на последующие проекты компании. Если производитель имеет возможность получить такие данные, проанализировать их и реализовать в следующих проектах те характеристики, которые позволят избежать аналогичных проблем для нового изделия, он не только сэкономит свои затраты на

послепродажное обслуживание, а сделает продукт, который лучше удовлетворит запросы клиента. А это уже влияет на имидж и конкурентоспособность производителя.

С помощью PLM заказчики получают возможность декларировать свои требования по улучшению продукта или претензии, связанные с ремонтом, которые будут непосредственно учтены конструкторами при проектировании следующей версии изделия. Таким образом, система PLM обеспечивает всем участникам жизненного цикла, включая и внешних, доступ к данным в реальном времени, реализуя сквозное распространение изменений, внесенных на одном из этапов, на все этапы жизненного цикла. Например, по требованию заказчика одна из деталей изделия будет крепиться в четырех точках вместо трех. Это требование повлечет изменения в спецификации детали и в настройке технологического процесса у поставщика для производства этой детали. В общей спецификации изделия изменится число болтов, необходимых для крепления такой детали, а это в свою очередь повлечет за собой изменения в заказе на закупку и т.д.

В целом, преимущества, которые дает PLM-решение, можно сформулировать следующим образом:

- ускорение вывода новой продукции на рынок, благодаря привлечению к процессам проектирования в реальном времени всех заинтересованных участников, включая внешних поставщиков и заказчиков;
- совершенствование характеристик разрабатываемой продукции и повышение качества, обнаружение недостатков и ограничений проекта на самых ранних стадиях;
- увязка проектирования и производственных процессов: инженеры-технологи становятся интегральной частью команды проектировщиков, благодаря чему проект сразу создается с учетом специфики производственного процесса, включая тестирование, контроль качества и т.д.;
- учет и использование опыта других проектов;
- реализация новой бизнес-модели «виртуального предприятия» — к процессу проектирования и производства привлекаются поставщики, либо работы определенного этапа жизненного цикла продукции передаются на аутсорсинг внешним компаниям.

## Производители

В центре объединения различных этапов жизненного цикла, которое реализует система PLM, пока остается ключевой для производства процесс проектирования, поэтому легко объясним тот факт, что первопроходцами современного рынка PLM стали производители САПР и систем PDM: PTC, EDS/SDRC, IBM/Dassault Systems [2]. Аналитики AMR Research выделяют эти компании в группу так называемых САПР-центричных поставщиков решений класса PLM [3]. Другая категория поставщиков в классификации AMR — это производители ERP-систем, такие как SAP, Oracle, Baan. Осознав стратегическое значение для своих заказчиков интеграции всех процессов по выпуску продукции на базе мощной информационной системы, эти компании стали активно предлагать собственные PLM-решения.

Технологические корни систем представителей этих двух групп определяют и основные различия между ними. «САПР-центричные» компании сильны в обеспечении наиболее неформальной, творческой части жизненного цикла изделия — многоитерационного проектирования в среде совместной работы над неструктурированными данными различной степени сложности. На базе этого ядра строятся взаимосвязи остальных этапов

жизненного цикла с центральным процессом проектирования, позволяющие понять, какое влияние оказывает проект на производство, определение требований к поставщикам, анализ предпочтений клиента, послепродажное обслуживание, и вывести обратные зависимости. Производства, где быстрые и эффективные модификации и создание новых изделий имеют критическое значение, такие как автомобильная промышленность и авиастроение, составляют основную клиентскую базу поставщиков PLM-решений из этой группы.

Традиционная сфера применения ERP-систем — бизнес-транзакции над структурированными данными, планирование ресурсов, контроль финансовых потоков, управление поставками, обеспечение информации для поддержки управленческих решений, контроль за выполнением плана. Свои PLM-решения компании «ERP-центричной» группы в основном строят вокруг структурированной спецификации изделия (BOM), на основе которой интегрируются процессы управления проектами, контроля за конфигурацией изделия, документооборота, управления потоком работ, управления цепочками поставок и взаимоотношениями с клиентами. Получающие сейчас распространение в ряде производственных компаний процессы разработки на заказ (engineer-to-order, ETO) и конфигурирования на заказ (configure-to-order, CTO), которые требуют управления структурой и конфигурацией изделия на протяжении всего жизненного цикла — оптимальная сфера применения таких PLM-систем.

AMR выделяет еще одну группу игроков на рынке PLM — независимые компании, не имеющие серьезного «веса» ни в сфере САПР, ни в области автоматизации общего управления корпоративными ресурсами — это Agile Software, MatrixOne, Eigner. Их решения имеют развитую функциональность для различных стадий жизненного цикла изделия за рамками трехмерного проектирования, включая поддержку взаимосвязей с внешними партнерами, и интерфейсы для интеграции в среду PLM необходимых САПР и ERP-приложений.

Задачи интеграции разных систем вообще актуальны для современного рынка PLM. На больших проектах может понадобиться комбинация возможностей, как это произошло, например, при реализации программы по созданию аэробуса A380 в компании Airbus, где решения SAP используются совместно с функциональностью систем PTC и IBM/Dassault. У производителя и поставщика могут уже быть установлены модули, на базе которых возможно построение общей PLM-среды. Другой вариант — производитель хочет дополнить имеющиеся у него модули автоматизированного проектирования и управления данными об изделии функциями управления взаимоотношениями с поставщиками и заказчиками, обратясь к решениям другого поставщика. Кроме того, на западном рынке есть целый ряд небольших компаний, производящих продукты, реализующие отдельные функции для управления жизненным циклом изделия, такие как моделирование материалов, 3-D анимация, управление технологическими процессами, поддержка среды совместной работы проектировщиков, тестирование готового изделия, автоматизация закупок, управление инженерными проектами, поддержка технического обслуживания и т.д. Они привлекательны для многих клиентов, поскольку приобретение полномасштабной платформы PLM требует первоначальных затрат в несколько миллионов евро, поэтому компании часто предпочитают «кусочную» автоматизацию, постепенно объединяя модули, приобретенные для решения частных задач. Одна из важных проблем формирующейся индустрии PLM — это стандартизация, возможно, создание в стеке программного обеспечения PLM общего уровня взаимодействия, позволяющего клиентам интегрировать на некую единую платформу оптимальные для себя решения по поддержке разных стадий жизненного цикла продукта.

Сложность PLM-решений как таковых, а также необходимость их интеграции в корпоративную информационную среду клиента, которая может включать самые разные системы — от собственных разработок и унаследованных приложений до последних новинок в области SCM и CRM, объясняет динамику развития профессионального сервиса на рынке PLM.

## Потребители

Решения PLM выросли на почве создания расширений к существующим САПР для построения сред совместной работы проектировщиков, поэтому сегодня наибольший процент использования возможностей PLM принадлежит инженерной сфере (рис. 2). Около 75% рынка PLM в 2002 году приходилось на предприятия автомобильной промышленности, ИТ-индустрию, самолетостроение и машиностроение. Эти дискретные производства имеют опыт не одного десятилетия по использованию САПР и PDM-систем, и потому вполне логичен их интерес к новому поколению решений по управлению данными об изделии. Кроме того, эти четыре отрасли в мире представляют уже вполне зрелые рынки, на которых необходимо искать особые пути выделиться среди конкурентов. Они уже миновали стадию ценовых войн, и теперь на первый план выходят такие параметры, как качество выпускаемой продукции, качество и высокий темп инноваций, эффективность поддержки полного жизненного цикла продукта. Но вслед за ними преимущества PLM-решений начинают по достоинству оценивать и другие отрасли, где необходимо серьезное внимание к стадии проектирования и поддержка достаточно длительного жизненного цикла производимой продукции: производство потребительских товаров и бытовой электроники, строительство, фармацевтика. При этом более 80% рынка PLM остается за дискретными производствами.



Рис. 2. Распределение прибыли от PLM-решений по функциональности систем

Первыми пользователями решений PLM были крупные компании (более 5 тыс. сотрудников) — одной из основных причин является накопленный опыт работы в среде САПР/PDM, ну и, конечно, материальные возможности. Именно их вложения позволяют рынку PLM-решений развиваться в сторону создания готовых пакетов PLM-приложений и постепенного снижения цен, что открывает перспективы использования PLM для средних (от 500 до 5 тыс. сотрудников) и небольших (от 20 до 500 сотрудников) компаний. За повышением интереса к PLM со стороны предприятий этого уровня лежат определенные тенденции в современном производственном бизнесе. Как правило, большинство таких предприятий являются поставщиками для крупных производителей, и потому нуждаются в механизмах увязки своих процессов проектирования и производства компонентов с теми целями, которые ставит «старший» партнер. Более того, основной производитель все чаще передает на аутсорсинг те или иные операции по разработке и сборке компонентов или даже сборке целого изделия. Такая бизнес-модель «виртуального» предприятия требует эффективной автоматизированной среды совместной работы, которую и обеспечивают системы PLM. Однако эта новая категория клиентов более сдержана в своих расходах и более подвержена влиянию экономической нестабильности, поэтому, чтобы иметь успех в этом сегменте рынка, поставщикам потребуется приложить больше усилий по формированию предложений, в которых будет найдено оптимальное для средних предприятий сочетание цены, функциональности, возможностей быстрой инсталляции и простоты в использовании.

## Что влияет на темпы распространения PLM-решений

Несомненно, залогом успешного развития и роста рынка PLM является оздоровление глобальной экономической ситуации, которое будет способствовать появлению у предприятий более широких возможностей для инвестиций в перспективные технологии. Аналитики выделяют еще несколько важных факторов, которые оказывают влияние на темпы адаптации PLM-решений клиентами.

**Проблема руководства.** Полномасштабная инсталляция PLM охватывает практически все подразделения предприятия — конструкторское, инженерное, производственное, отдел закупок, службу контроля качества, маркетинг и продажи, техническое обслуживание. Только руководители верхнего звена имеют достаточные полномочия, чтобы осуществить проект, затрагивающий интересы фактически всех сотрудников организации, однако пока общепринятая практика — возлагать эту задачу на менеджеров инженерных подразделений.

**Необходимость преодолевать общекультурные сложности.** PLM ставит в центр всех процессов на производственном предприятии создаваемый продукт (вместо бухгалтерских документов, например) и позволяет разделять данные об изделии специалистами разного уровня и квалификации. Возможно, кто-то из сотрудников потеряет осознание собственного исключительного положения, поскольку теперь владение информацией по проекту — не только его привилегия. Несмотря на то что в соответствии с философией PLM процесс проектирования ставится во главу угла в бизнесе производственной компании, нередко приходится сталкиваться с негативным отношением к PLM именно со стороны конструкторских отделов. Благодаря PLM проект становится открытым для других функциональных подразделений и внешних партнеров, и этот факт вызывает у создателей проекта в лучшем случае скептицизм, в худшем — явное неприятие.

Проблемы интеграции. Решения PLM должны сосуществовать с различными корпоративными системами, а также поддерживать интеграцию с решениями компаний-поставщиков и заказчиков предприятия.

**Формирование надежных механизмов для разделения интеллектуального капитала.** Информация, которая должна быть доступна всем пользователям интегрированной среды PLM, в том числе проектные данные, данные анализа рынка, инновационные прогнозы, уникальные клиентские требования к разрабатываемому изделию являются интеллектуальной собственностью компании. Поэтому в рамках проекта PLM равно необходимы как средства защиты на уровне технологий, так и способы установления доверительных отношений между партнерами.

**Формирование репозитория PLM как основного источника достоверных данных о продукте для всех категорий пользователей.** Большинство преимуществ использования PLM вытекает из кумулятивных возможностей, которые предоставляют эти системы. При создании общекорпоративного репозитория необходимо обеспечить соответствующие функции электронного хранилища, реализовать средства каталогизации, поиска, разделения данных и визуализации, причем так, чтобы они удовлетворяли разные категории пользователей. Время и усилия потребуются не только для того, чтобы решить эти задачи, но и чтобы довести их преимущества до тех, кому они предназначены.

**Стандартизация.** В области PLM стандартизация необходима как в сфере представления данных частными САПР и PDM-системами, так и для создания общего синтаксиса для

совместного использования данных модулями разных PLM и другими корпоративными системами. Эволюция и адаптация стандартов в этой области, как и в любой другой, требует времени.

Общекорпоративный масштаб PLM-решений (в идеале) ставит и серьезные проблемы их внедрения. Поставщики и консультанты по PLM сегодня стремятся избежать неприятностей, которые почти обязательно сопровождают развертывание решений аналогичной сложности, например, ERP: затягивание сроков проекта, постоянный рост расходов на реализацию и неудовлетворенность заказчика темпами возврата инвестиций. Поэтапная реализация PLM-проектов помогает ускорить окупаемость и получать реальный эффект от внедрения уже через полгода с начала реализации.

Но, несмотря на все трудности, современные производственные компании понимают, что необходимое условие для стабилизации и развития бизнеса — совершенствование всех процессов, связанных с созданием продукции. Чтобы стать известным или удержать репутацию своей марки, надо обеспечивать высокое качество проектирования и производства. А сложные экономические условия подчас только подстегивают: клиенты в это время становятся более разборчивыми.

■ Agile Software	■ EDS PLM	■ Pelion Systems	■ SiteScape
■ Alventive	■ Framework Technologies	■ PeopleSoft PLM	■ Smarteam
■ Axeda	■ GlobalSpec	■ Primavera	■ SolidWorks
■ Bentley Systems	■ HandySoft	■ Proficiency	■ Sopheon
■ BetaSphere	■ iBaan PLM	■ PTC	■ Synchronicity
■ BOM.COM	■ IBM PLC	■ QSA	■ Think3
■ Centric Software	■ MatrixOne	■ Rely Software	■ TightLink
■ CoCreate	■ MS2	■ SAP PLM	■ TogetherSoft
■ DataSweep	■ NetVendor	■ SDRC	■ Zweave
■ Dessault Systemes	■ Oracle PLM	■ Silvon	

Поставщики PLM-решений

## Литература

1. Aberdeen Group, Worldwide PLM Spending (from the engineering perspective). Forecast and analysis 2001-2005. 2002 December.
2. В. Климов, В. Краюшкин, М. Пирогова, Настоящее и будущее PDM. // Открытые системы, 2002, № 2.
3. M.Burkett, ERP will never meet all the PLM requirements of innovative manufacturers. AMR Research, 2003 February.

---

По данным Aberdeen Group [1] объем рынка PLM-решений к 2005 году составит 5,12 млрд. долл. и его ожидает ежегодный рост, несмотря на стагнацию мировой экономики. К традиционным пользователям — крупным производственным корпорациям — начинают присоединяться средние компании, и наряду с США активно осваивать PLM-решения продолжают Европа и Азия.

\*\*\*

Выстроить в одной системе всю цепочку процессов жизненного цикла, обеспечить их взаимосвязь и разделение информации на разных стадиях работы над продуктом позволяет централизованный репозиторий, в котором система PLM сохраняет данные, полученные на каждом этапе.

\*\*\*

По данным Aberdeen, в мире более 50% затрат на PLM-решения приходится на оплату услуг консультантов. Аналитики также прогнозируют повышение интереса к рынку PLM ведущих производителей аппаратного обеспечения, которые могут использовать возможность поставки специально сконфигурированных серверных платформ вместе с приложениями PLM как шанс придать новый импульс продажам своей техники.

#### 4. Работа в реальном времени.

##### *Запуск MPB*

MPB состоит из двух компонентов: **сервер матообработки** и **графическая консоль**.

**Сервер матообработки** реализует функции пересчета базы каналов, обработки данных и управления, связи с контроллерами и удаленными мониторами по сети, последовательному интерфейсу и через модемы. Кроме того, здесь же реализуются функции обмена с базами данных и другими приложениями WINDOWS, а также архивирования и считывания данных из архивов.

Функции человеко-машинного интерфейса реализуются в **графической консоли**. Этот модуль запрашивает данные у сервера матообработки для их отображения и передает обратно команды управления, сформированные оператором. Для связи сервера матообработки и графической консоли используется механизм DCOM.

Модуль графической консоли может запускать сервер матообработки. Однако он может быть запущен независимо. В этом случае консоль устанавливает связь с запущенным сервером.

##### **Запуск графической консоли**

Графическая консоль запускается следующей командной строкой (директория инсталляции модуля должна быть текущей):

```
PicRT.exe [<prg>.ctm /N:<node>] [/S:<PC>] [/F] [/R] [/H]
```

где

<prg> – имя файла конфигурации проекта;

<node> – имя узла;

<PC> – имя компьютера, на котором должен работать сервер матообработки. Отсутствие этого параметра означает локальный компьютер;

/F – выход при старте в полноэкранный режим;

/R – автоматический запуск сервера матообработки при наличии пользователя с именем **default** и паролем **default**. Если его нет, то на экран выводится диалог запроса имени и пароля;



**/H** – при задании этого ключа на экран не выводится инструментальная панель. Ключ работает только при наличии ключей **/R** и **/F** и игнорируется при отсутствии хотя бы одного из них.

Все параметры запуска являются необязательными. Их можно указать после запуска исполнительного модуля **PicRT.exe**.

**Внимание!** Графическая консоль может запустить сервер матобработки только на локальном компьютере. При подключении к удаленному компьютеру, сервер матобработки должен быть на нем уже запущен.

### Запуск сервера матобработки

Сервер матобработки может быть запущен без использования графической консоли следующей командной строкой (директория инсталляции модуля должна быть текущей):

```
DrawServ.exe /P:<path> /F:<node> [/RUN] [/CONSOLE] [/AUTORS] [/IREC=n] [/REC=m]
[/DEBUG=h] [/DISABLE_IO] [/I:NNNN] [/DISK=i] [/BUFFS=d]
```

где

**<path>** – полный путь к базе каналов;

**<node>** – имя базы каналов без расширения.

**/RUN** – запуск пересчёта при старте;

**/CONSOLE** – вывод на экран окна с таблицей каналов.

**/IREC=n** – число NCB для индивидуального приема n=0,1,2 (по умолчанию 1);

**/REC=m** – число NCB для приема включая IREC

**/DEBUG=h** – вывод отладочной информации в файл **<name>.tnt**, где **<name>** – имя файла базы каналов. Этот ключ реализуется только для профайлера. Параметр **h** – это число в шестнадцатеричном формате, каждый бит которого указывает на сохранение определенного вида информации:

**1 бит** – информация о каналах;

**2 бит** – информация о загружаемых объектах;

**3 бит** – перечень имен и индексов FBD-программ, описанных в узле;

**4 бит** – перечень используемых IL- и FBD-программ с числом их вызова;

**5 бит** – перечень метапрограмм;

**6 бит** – проверка аргументов FBD-программ при запуске;

**7 бит** – информация по сетевым послылкам;

**8 бит** – информация о резервировании узлов;

**10 бит** – дополнительная информация об обмене по последовательным портам.

**11 бит** – имя и номер канала (заданного каналом [Отладка](#)) при каждом вызове им FBD-программы;

**13 бит** – при вызове каналом (заданным каналом [Отладка](#)) FBD программы в файл выводятся входные и выходные аргументы, которым было произведено присвоение, а также по каждому FBD блоку его номер, тип, значения входов и выходов;

**14 бит** – при изменении любого атрибута канала, указанного каналом [Отладка](#), в файл записывается следующая строка:

CCD: <инициатор> [параметр] (индекс атрибута) <значение атрибута>

где

- <инициатор> – инициатор изменения (сеть, RS, формы отображения, DDE, OPC, ODBC, FBD);

- <параметр> – дополнительная информация об инициаторе изменений (например, номер FBD-программы, идентификатор пользователя).

**15 бит** – уведомление графических консолей.

**/DISABLE\_IO** – замена каналов обмена с платами УСО на внутренние каналы:

**АНАЛОГ INPUT** на канал [G\\_пила](#) подтипа ПУСТОЙ;

**АНАЛОГ OUTPUT** на канал [out\\_null \(F\)](#) подтипа ПУСТОЙ;

**ДИСКРЕТ INPUT** на канал [G\\_бегущая\\_1](#) подтипа ПУСТОЙ;

**ДИСКРЕТ OUTPUT** на канал [out\\_null \(H\)](#) подтипа ПУСТОЙ;

**/AUTORS** – этот ключ определен для каналов [подтипа DCS](#), [подтипа MODBUS](#), а также для каналов подтипа **СВЯЗЬ** с дополнениями [In M\\_Link](#), [Out M\\_Link](#) :

– профайлер:

если каналы настроены на COM1 (COM2) и его не удалось инициализировать при запуске, то эти каналы перенастраиваются на COM2 (COM1) с теми же параметрами. Порт, на который производится перенастройка, должен быть задан в бланках настройки узла, но каналов, настроенных на него, быть не должно.

– профайлер и MPB:

при наличии ключа **AUTORS** и при задании нескольких COM-портов в бланках настройки узла: если настройка каналов **RS** = 255, то эти каналы при запуске настраиваются на порт, который первым инициализируется без ошибок.

**/BUFFS=d** – число буферов для поднятия архива (по умолчанию – 16). Один буфер – 128 записей;

**/DISK=i** – индекс диска, на котором сохраняется локальный отчет тревог (1 – A, 2 – B, 3 – C, 4 – D);

**NNNN** – число в шестнадцатеричном формате. Значение его отдельных битов задает следующие параметры:

**1,2 биты** – ограничение числа NCB на прием:

1 – 3 блока;

2 – 6 блоков;

3 – к рассчитанному числу добавить 4 блока;

**3,4 биты** – ограничение числа NCB на отсылку:

1 – 4 блока;

2 – 6 блоков;

3 – 8 блоков;

**5 бит** – запрет считывания границ каналов из файла сохранения состояния;

**6 бит** – установить TIME\_CRITICAL приоритет потоку пересчета базы каналов;

**7 бит** – запретить доступ к базе каналов по DDE, NetDDE;

**8 бит** – установить приоритет LOWEST потоку DDE;

**9 бит** – не использовать групповое сетевое имя;

**15,16 бит** – время оповещения графики о пересчете:

0 – при периоде пересчета больше 2000 мс оповещать 1500 мс, а если меньше 100 мс, то оповещать через 100 мс;

1 – 950 мс;

2 – 50 мс;

3 – 5000 мс.

**Внимание!** Чтобы подключить к серверу матобработки несколько графических консолей с удаленных компьютеров, его следует запустить с командной строки, а не с помощью локальной графической консоли.

## Настройка DCOM

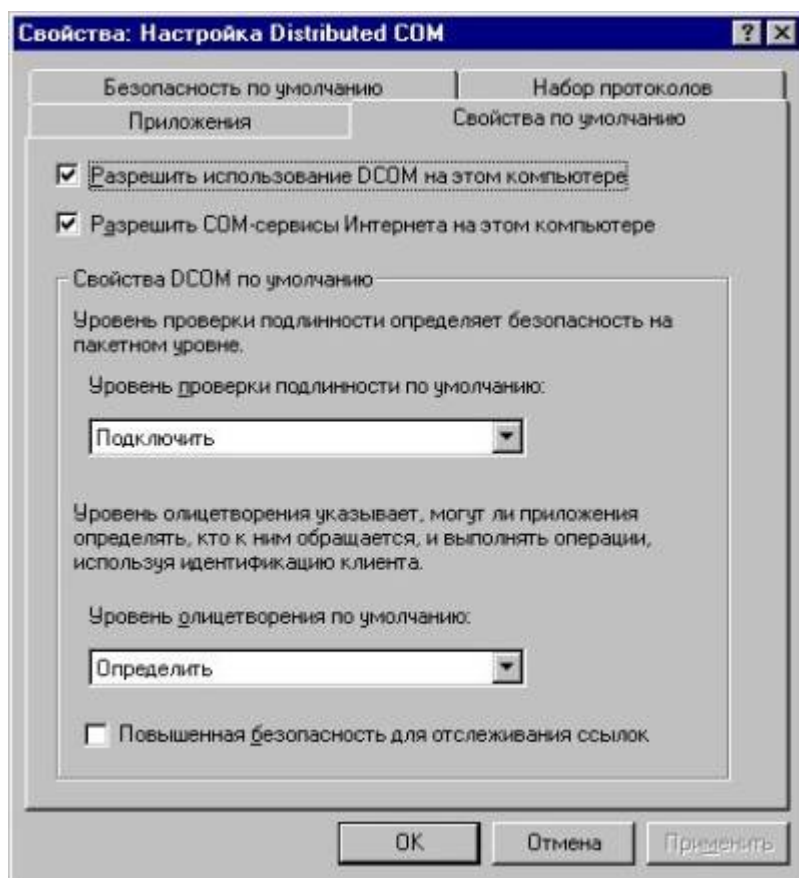
Взаимодействие математической и графической компонент MPB осуществляется через DCOM. Этот механизм позволяет запускать графическую и математическую компоненты MPB на разных компьютерах, объединенных в локальную сеть.

Для запуска MPB под WINDOWS 95 поддержку DCOM необходимо установить.

Для запуска компонент MPB на разных компьютерах необходимо осуществить дополнительную настройку DCOM. Для этого надо сначала зарегистрировать сервер матообработки на обоих компьютерах. При инсталляции MPB эта регистрация осуществляется автоматически. Однако при переносе сервера матообработки с одного компьютера на другой его регистрацию можно осуществить вручную. Это делается с помощью программы **tmreg.exe**, которая размещается в директории инсталляции системы.

**Внимание!** Если используется одноранговая сеть, то для работы DCOM учетные записи пользователей на всех машинах должны быть одинаковыми.

После регистрации сервера матообработки следует запустить программу **Dcomcnfg.exe** из поддиректории **SYSTEM32** директории установки WINDOWS NT. При этом на экран будет выведен диалог **Свойства: Настройка DCOM**. В этом диалоге надо войти в бланк **Свойства по умолчанию** и настроить свойства



При установлении связи между Windows 2000 и Windows98/95 значения настроек "Уровень проверки подлинности по умолчанию" и "Уровень олицетворения по умолчанию" должны быть соответственно "Нет" и "Аноним".

Кроме того, используя бланк **Безопасность по умолчанию** диалога настройки **DCOM**, нужно задать соответствующие разрешения на доступ к серверу для удаленных пользователей.

Для подключения нескольких графических консолей к одному серверу матобработки эту настройку надо выполнить на всех компьютерах, где будут запускаться графические консоли. При этом регистрация сервера матобработки обязательна на всех этих компьютерах.

## Структура обработки данных

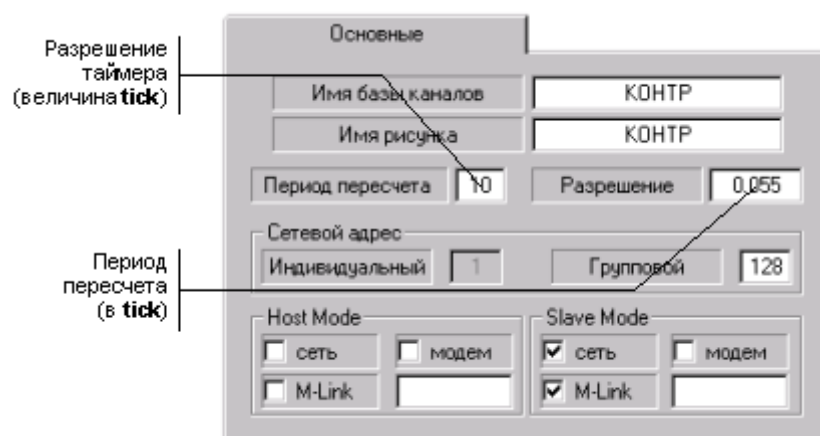
Прикладные системы, созданные в инструментальной системе ТРЕЙС МОУД, запускаются под управлением мониторов исполнительной системы. Они циклически выполняют следующие операции: обмен данными с контроллерами и УСО, сохранение данных в СПАД и ОТ, пересчет базы каналов, обмен по сети, взаимодействие с графическими клиентами, обмен данными с другими приложениями и др. Набор операций зависит от типа монитора. Так, например, Микро МРВ не поддерживает архивирование, обмен с другими приложениями и графическими консолями.

### Пересчет базы каналов

Мониторы реального времени ТРЕЙС МОУД работают как интерпретаторы базы каналов. Интерпретация базы каналов осуществляется один раз за цикл системы.

### Условие пересчета базы каналов

Условием очередного пересчета базы каналов является начало нового цикла системы. Время цикла настраивается индивидуально для каждого узла с помощью двух параметров, задаваемых в соответствующих областях бланка **Основные** диалога **Параметры узла** в редакторе базы каналов: Это – **период пересчета** в tick и **разрешение** таймера (величина tick). Разрешение таймера задается в секундах и по умолчанию равно 0,055 с.



**Внимание!** Для Микро МРВ нельзя задавать разрешение таймера больше 0.055 секунды.

Время цикла может быть немного больше установленного значения. Это объясняется влиянием многозадачности Windows.

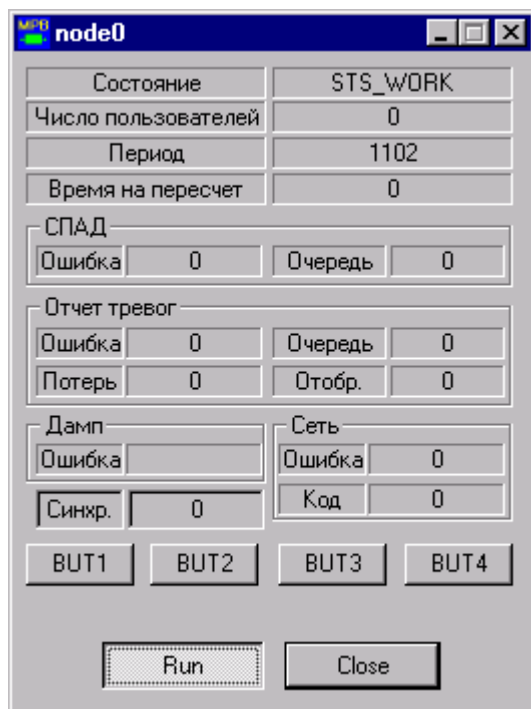
Контролирует переход к новому циклу канал **СИСТЕМНЫЙ** с дополнением [Индекс пересчета](#) .

Величина цикла определяет минимальное время реакции системы.

### Контроль временных параметров

Контроль временных параметров работы монитора можно осуществлять в диалоге сервера матообработки или с помощью ряда каналов.

Диалог сервера матообработки показан на следующем рисунке.



Здесь выводится информация, характеризующая текущее состояние выполняемых им задач. Этот диалог доступен для просмотра только при запуске сервера с командной строки. В нем выводится следующая информация:

**Состояние** – состояние сервера матообработки. Его можно контролировать каналом **СИСТЕМНЫЙ** с дополнением [Статус](#) :

- 0 - STS\_NOTWORK – сервер не работает;
- 1 – STS\_START – сервер запускается;
- 2 – STS\_WORK – нормальная работа;
- 3 – STS-TRACE – подчиненное состояние;
- 4 – STS\_WAIT – состояние ожидания;
- 5 – STS\_STOP – сервер останавливается;
- 6 – STS\_ERROR – ошибка в работе сервера;

7 – STS\_NORMA – нормальное завершение.

**Число пользователей** – число подключенных пользователей. Каждый вход пользователя в систему добавляет 1 к этому счетчику, каждый выход пользователя - уменьшает его на 1.

**Период** – текущее время цикла сервера матообработки в миллисекундах. Его можно контролировать каналом подтипа **ДИАГНОСТИКА** с дополнением **Цикл (реально)** .

**Время на пересчет** – время, затраченное на последний пересчет базы каналов в миллисекундах. Контролируется каналом подтипа **СИСТЕМНЫЙ** с дополнением **Время пересчета** .

Раздел **СПАД** – состояние работы со СПАД и длина очереди для сохранения в СПАД. Эти параметры контролируются каналами **ДИАГНОСТИКА** с дополнением **СПАД** и дополнением **Очередь СПАД** соответственно.

Раздел **Отчет тревог** – состояние работы с отчетом тревог, длины очередей для сохранения в него и передачи в графические консоли, а также количество потерянных сообщений. Эти параметры контролируются каналами **ДИАГНОСТИКА** с дополнениями **Отчет тревог** , **Очередь ОТ** , **Очередь (отображение)** , **Потеря ОТ** соответственно.

**Дамп** – характеристика сохранения состояния системы, контролируется с помощью канала **ДИАГНОСТИКА** с дополнением **Дамп** .

**Сеть** – состояние сетевого обмена и код, возвращаемый сетевой операционной системой. Эти параметры доступны для контроля с помощью каналов **ДИАГНОСТИКА** с дополнением **Сеть** и дополнением **Код сети** соответственно.

**Синхр.** – в этом поле отображается значение канала подтипа **ДИАГНОСТИКА** с дополнением **Рассинхронизация** .

В нижней части диалога сервера матообработки имеются четыре кнопки. Нажатие ЛК на каждой из них инвертирует соответствующий бит в значении, контролируемом каналом **СИСТЕМНЫЙ** с дополнением **Код клавиши** .

Канал **ДИАГНОСТИКА** с дополнением **Превышение** контролирует количество ситуаций, когда реальное время пересчета базы каналов превышало установленный период пересчета. Если для этого канала установить тип OUTPUT, то любая его отработка сбрасывает значение счетчика превышений на 0.

Две нижние кнопки в диалоге – **Run** и **Close** – управляют запуском пересчета базы каналов и завершением работы сервера матообработки соответственно.

### **Такты пересчета**

Один пересчет базы каналов включает в себя четыре такта:

пересчет всех каналов типа **INPUT** кроме каналов подтипов **КАНАЛ** и **ОБЪЕКТ**. При этом для каждого пересчитываемого канала последовательно выполняется **трансляция** входных значений в аппаратные и реальные и процедура **управление**;

пересчет каналов типа **INPUT** подтипов **КАНАЛ** и **ОБЪЕКТ** (для каждого пересчитываемого канала последовательно выполняется **трансляция** входных значений в аппаратные и реальные и процедура **управление**). Выполнение процедуры **управление** для всех каналов, пересчитываемых на этом цикле;

вычисление метапрограмм, написанных на Техно IL;

пересчет каналов типа **OUTPUT** (**трансляция** входных значений в реальные и аппаратные).

Один цикл пересчета включает в себя три прохода по базе каналов, начиная с канала, имеющего младший индекс. Эти проходы реализуются на первом, втором и четвертом тактах пересчета.

### **Особенности пересчета каналов**

[Пересчет значений канала типа INPUT](#)

[Пересчет значений канала типа OUTPUT](#)

#### **Пересчет значений канала типа INPUT**

При наличии у канала **INPUT** признака аппаратной недостоверности при выполнении остальных условий не выполняется только процедура **трансляция**. Соответственно, его реальное значение не изменится. Однако процедура **управление** для этого канала выполняется.

#### **Пересчет значений канала типа OUTPUT**

При наличии у канала **OUTPUT** признака аппаратной недостоверности на каждом цикле пересчета его выходное значение будет отсылаться приемнику данных. Эти попытки продолжаются, пока не сбросится признак недостоверности или канал не будет выключен.

**Внимание!** Канал **OUTPUT** обрабатывает свою функцию (передает приемнику значение своего атрибута **ВЫХОД** или исполняет свою системную команду) только при изменении значения своего атрибута **ВЫХОД**. Поэтому для инициализации принудительного исполнения им своей функции, если его **АППАРАТНОЕ** значение не изменялось, достаточно изменить значение атрибута **ВЫХОД**. При этом для отработки будет использовано текущее значение атрибута **АППАРАТНОЕ**. Заметим, что такая процедура, примененная к каналам, функции которых реализуются асинхронными параллельными потоками, может привести к неоднозначному результату в случае, если предыдущая команда еще не была исполнена.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 11. [Работа в реальном времени Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Работа в реальном времени».**



## ЛЕКЦИЯ 12 НАДЕЖНОСТЬ SCADA – СИСТЕМ

1. Архитектура клиент-сервер
2. Дублирование сервера ввода-вывода
3. Резервирование на уровне задач
4. Выделенный сервер файлов
5. Резервирование сети
6. Резервирование связи с контроллерами
- 7.

### надежности SCADA – систем

Современные методы управления производственным процессом на основе компьютерных технологий получили широкое распространение на большинстве промышленных предприятий. Все успешно работающие системы обеспечивают контроль и управление, включая графический интерфейс оператора, обработку сигналов тревог, построение графиков, отчетов и обмен данными. В тщательно спроектированных системах эти возможности способствуют улучшению эффективности работы предприятия и, следовательно, увеличению прибыли. Однако, при разработке таких систем, инженеры часто упускают из вида один существенный аспект - что произойдет, если какой либо элемент аппаратуры выйдет из строя?

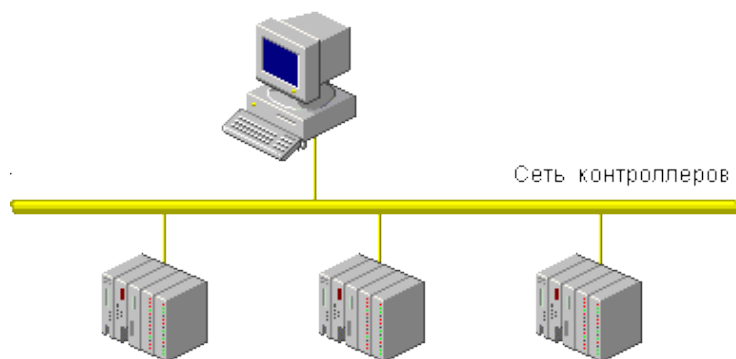


Рис. 8. Локальная система АСУТП

Локальная система АСУТП показанная на рисунке 8 и распределенная система на рисунке 9 имеют одну общую особенность. Обе системы полностью выйдут из строя, если всего в ОДНОМ компоненте системы (компьютере, соединенном с контроллерами или сетью контроллеров) возникнет неисправность.

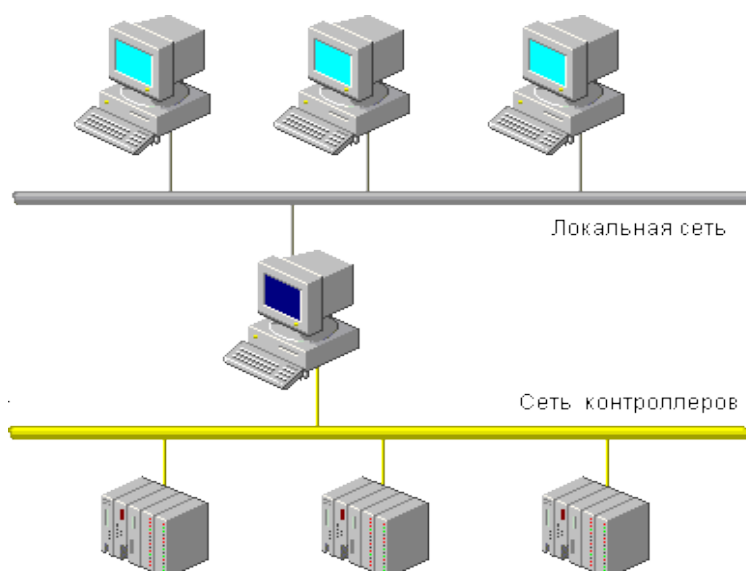


Рис.9. Распределенная система АСУТП

Большинство современных компьютеров обеспечивают хорошие показатели надежности, но тем не менее они также выходят из строя, особенно при эксплуатации в жестких производственных условиях. Если какие либо компоненты производственного процесса являются критически важными (или весь процесс), или стоимость остановки производства очень высока, возникает НЕОБХОДИМОСТЬ построения резервируемых систем. В системах обеспечивающий резервирование, выход из строя одного компонента не влечет за собой остановку всей системы. Программное обеспечение для управления производственными процессами (SCADA система) поддерживает реализацию резервирования большинства компонентов, как вследствие особенности архитектуры, так и наличие встроенных механизмов.

#### *Архитектура клиент-сервер*

Распределение процессов управления и контроля по нескольким компьютерам, объединенных в локальную сеть позволяет увеличить эффективность и скорость работы всей системы. В простой системе, компьютер соединенный с промышленным оборудованием, становится сервером, предназначенным для взаимодействия с контроллерами, в то время как компьютеры в локальной сети - клиентами ( Рис.10).

Когда компьютеру-клиенту требуются данные для отображения, он запрашивает их у сервера и затем обрабатывает локально.

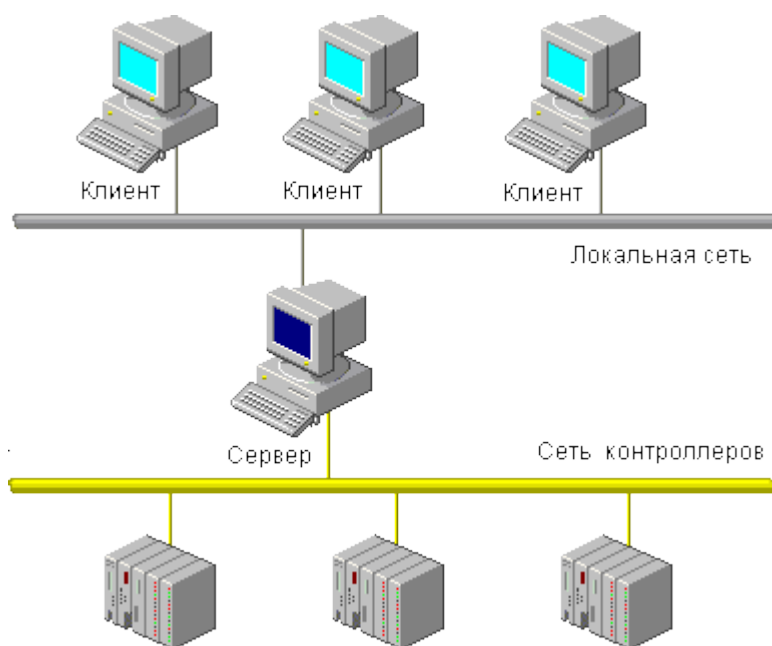


Рис.10. Клиент - серверная архитектура простой системы

### *Дублирование сервера ввода-вывода*

Для обеспечения резервирования в систему может быть добавлен второй (резервный) сервер, также предназначенный для взаимодействия с промышленным оборудованием (Рис.11).

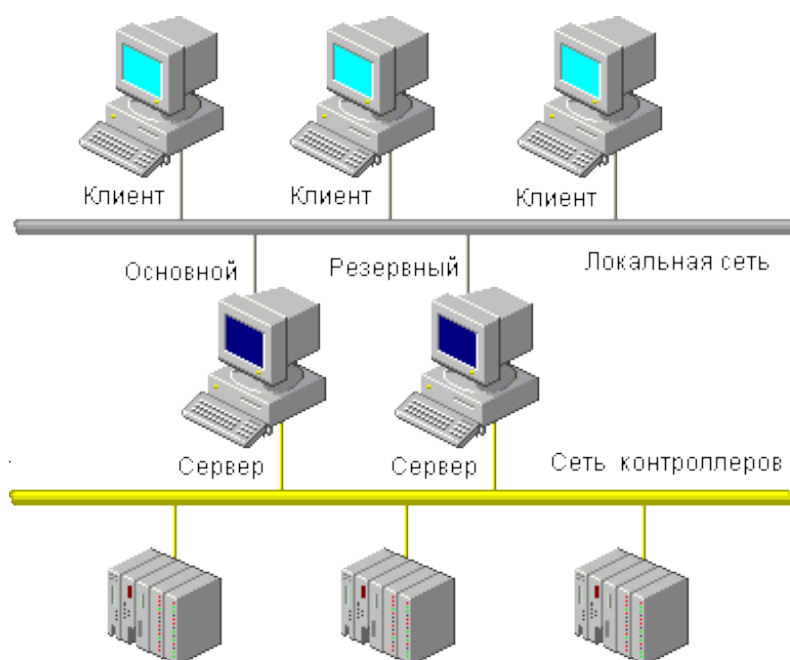


Рис.11. Система с дублированным сервером

Резервный сервер не должен при этом полностью дублировать работу основного, поскольку в этом случае оба сервера взаимодействуют с контроллерами, удваивая нагрузку на промышленную сеть, сокращая, следовательно, общую производительность. В клиент-серверной архитектуре только основной сервер взаимодействует с контроллерами. Одновременно он обменивается данными с резервным сервером, постоянно обновляя его статус. Если обмен данными с основным сервером прекращается, резервный сервер полагает, что основной вышел из строя и берет на себя его функции. После того, как неисправность в основном сервере будет устранена и он будет снова включен, основной сервер считает текущее состояние с резервного сервера и восстановит свою роль в качестве основного.

### *Резервирование на уровне задач*

В клиент-серверной архитектуре, при наличии дублированных серверов ввода-вывода можно реализовать более чем просто поддержку постоянной связи с промышленными устройствами. Необходимо также обеспечить сохранность и непрерывность данных тревог и графиков в случае возникновения неисправности. Это может быть обеспечено путем разделения функций сервера на 4 задачи:

- Ввод-вывод
- Тревоги
- Графики
- Отчеты

Каждая из этих задач поддерживает свою базу данных независимо от других задач, так что можно дублировать каждую задачу в отдельности. Например, можно обеспечить параллельное исполнение задач отображения графиков на разных серверах в отличие от архитектуры основной/резервный, используемой для серверов ввода-вывода.

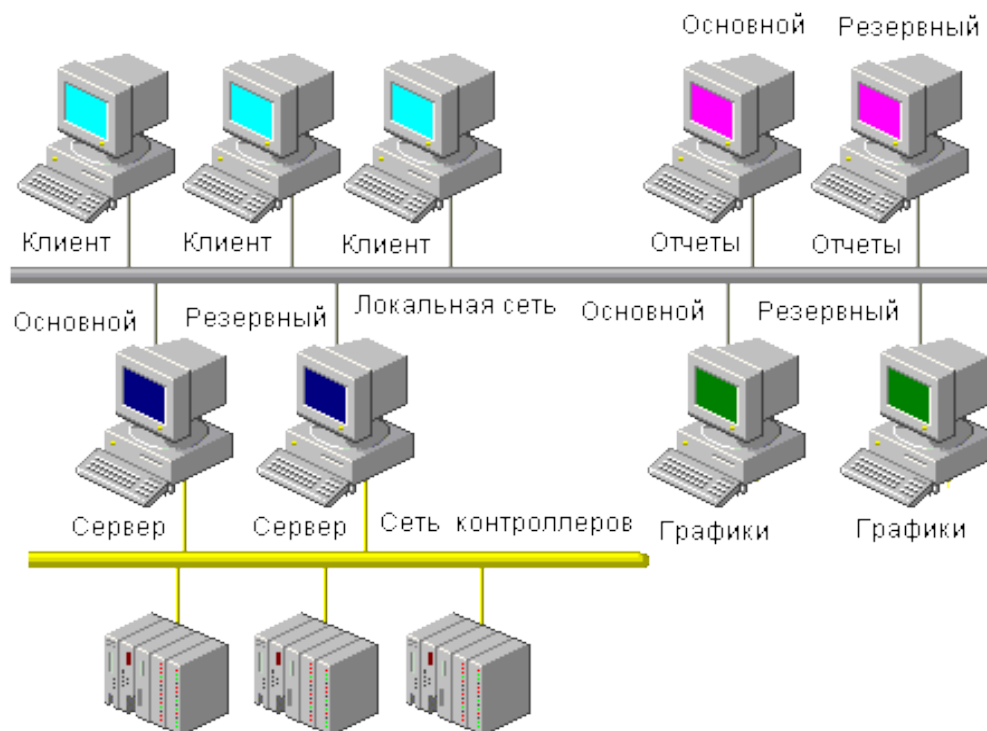


Рис. 12. Резервирование задач отображения графиков и вывода отчётов

Во время работы система обеспечивает параллельную работу основных и резервных серверов. Если основной сервер Отчетов, Графиков или Тревог выходит из строя, все

клиенты получают данные с резервного. После рестарта основного сервера клиенты сохраняют работу с резервным сервером до тех пор, если он не выйдет из строя или произойдет выключение и перезагрузка клиента. Поскольку система обеспечивает идентичность данных на обоих серверах, для клиента нет никакой разницы откуда брать данные, с основного или резервного сервера, ситуация когда часть клиентов берет данные с основного, а часть с резервного сервера, является нормальной. После устранения неисправности основного сервера он может обновить свои данные графиков с помощью информации с резервного. Таким образом, поддерживается непрерывное отображение информации графиков.

#### *Выделенный сервер файлов*

В систему может также быть добавлен выделенный сервер файлов для централизованного хранения баз данных и информации для отображения на экране. В случае выхода из строя основного сервера обеспечивается непрерывное отображение графиков. Централизованные базы данных также легче поддерживать и администрировать.

#### *Резервирование сети*

Структура, представленная на Рис.11, увеличивает надежность системы путем устранения "слабых" мест - в данном случае сервера ввода-вывода. Однако, если сеть выходит из строя, управление на клиентских компьютерах также нарушается. Дополнительная сеть и файловый сервер обеспечивает стабильность работы системы даже в случае выхода одной из сетей из строя. (Рис.13).

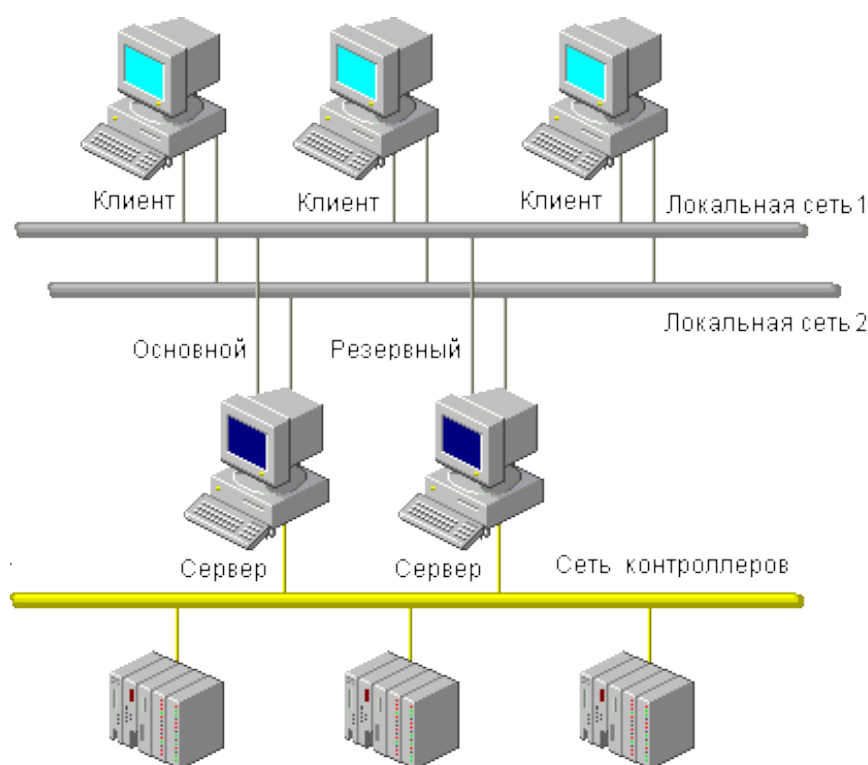


Рис. 13. Резервирование сети (каналов связи)

#### *Резервирование связи с контроллерами*

В большинстве контроллеров можно организовать дополнительную связь между сервером ввода-вывода и устройством. Наличие дополнительного канала связи гарантирует сохранение обмена данными, если основной канал выйдет из строя.



Рис. 14. Резервирование контроллеров

Во время старта система соединяется с устройством по основному каналу связи. Если обмен данными нарушается (например, обрыв кабеля), система переключается на резервный канал. Обратный переход на основной канал происходит после восстановления физического соединения. Резервный путь обмена данными можно также организовать по локальной сети, как показано на следующем рисунке:

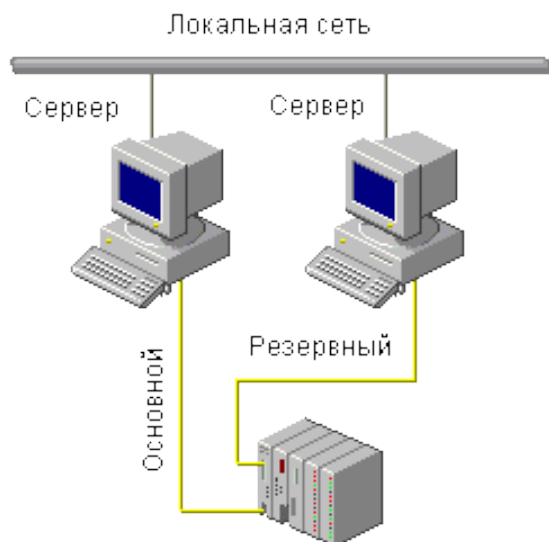


Рис.15. Резервирование обмена данными с помощью локальной сети

В этом примере взаимодействие с устройством ввода-вывода поддерживается непрерывным, даже если один из серверов или коммуникационных кабелей выйдет из строя.

Если устройство ввода-вывода поддерживает соединение точка-точка, можно обеспечить полное резервирование путем дублирования устройств (Рис.16).

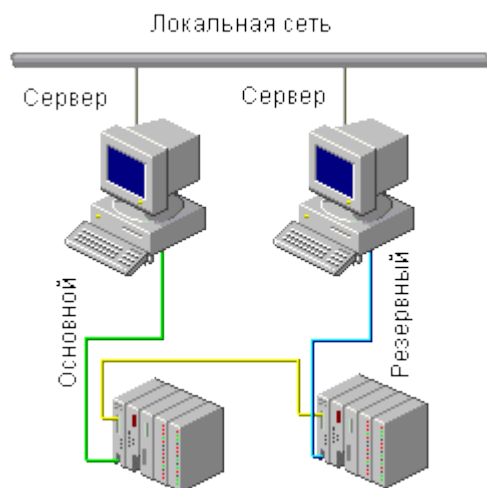


Рис. 16. Полное резервирование связи с контроллерами

Необходимо также отметить, что конкретная реализация всех вышеприведенных возможностей повышения надежности существенно различается в разных пакетах SCADA. Основным критерием можно считать простоту настройки реальных конфигураций, то есть программная поддержка решений, изначально заложенная в пакете. Все возможности по резервированию полностью встроены в пакет и не требуют никакого дополнительного программирования. Причем большая часть из них устанавливается с помощью интуитивно понятного "Мастера Настройки Компьютера" (Computer Setup Wizard).

В заключение нужно отметить, что постоянно возрастающая сложность систем АСУТП и необходимость их интеграции с корпоративными ИТ системами выдвигает требования по надежности и масштабируемости систем SCADA на первое место в длинном перечне остальных свойств пакетов данного класса.

## 7. Разработка распределенных систем.

### Обзор раздела

Этот раздел посвящен описанию возможностей и инструментов ТРЕЙС МОУД для создания распределенных систем управления и мониторинга.

Здесь рассматриваются особенности обмена по локальной сети, сети на базе последовательного интерфейса, базирующейся на протоколе M-LINK, и обмена по коммутируемым линиям с использованием модемов.

Этот раздел включает в себя следующие подразделы:

Идеология построения распределенных систем;

Сетевой обмен;

Обмен по протоколу M-LINK;

Обмен по коммутируемым линиям.

В них, кроме описания взаимодействия различных узлов проекта текущей версии, приводятся механизмы настройки связи с мониторами ТРЕЙС МОУД версии 4.20. Это позволит интегрировать уже существующие операторские станции и контроллеры, работающие под управлением ТРЕЙС МОУД предыдущей версии, в единую систему с новыми.

## **Идеология распределенных комплексов**

ТРЕЙС МОУД имеет мощные средства для создания распределенных АСУТП, включающих в себя до трех уровней в иерархии. При их построении могут использоваться локальные сети, сети на базе последовательного интерфейса RS-485, RS-422, радиомодемов и коммутируемых линий, GSM и InterNet.

Максимальное число серверов реального времени в одном проекте равно 128 (исключая серверы документирования, WEB-активаторы и GSM-активаторы). Кроме того, проект может включать в себя неограниченное количество архивных станций, работающих под управлением мониторов SUPERVISOR, и графических клиентов – мониторов NetLINK Light.

Серверы реального времени могут обмениваться по последовательному интерфейсу, используя протокол M-LINK. Для этого могут применяться стандартные интерфейсы, радиоканал, коммутируемые линии и GSM сети.

Надо заметить, что при разработке крупных сетевых систем, включающих в себя десятки узлов, лимитирующим элементом становятся не характеристики пакета по количеству одновременно работающих в сети узлов, а пропускная способность линий связи.

### ***Уровни АСУ***

[Уровень контроллеров](#)

[Оперативный уровень](#)

[Административный уровень](#)

Исполнительная система ТРЕЙС МОУД включает в себя мониторы, предназначенные для работы на всех уровнях систем управления. В рамках идеологии построения АСУТП в ТРЕЙС МОУД можно выделить три уровня:

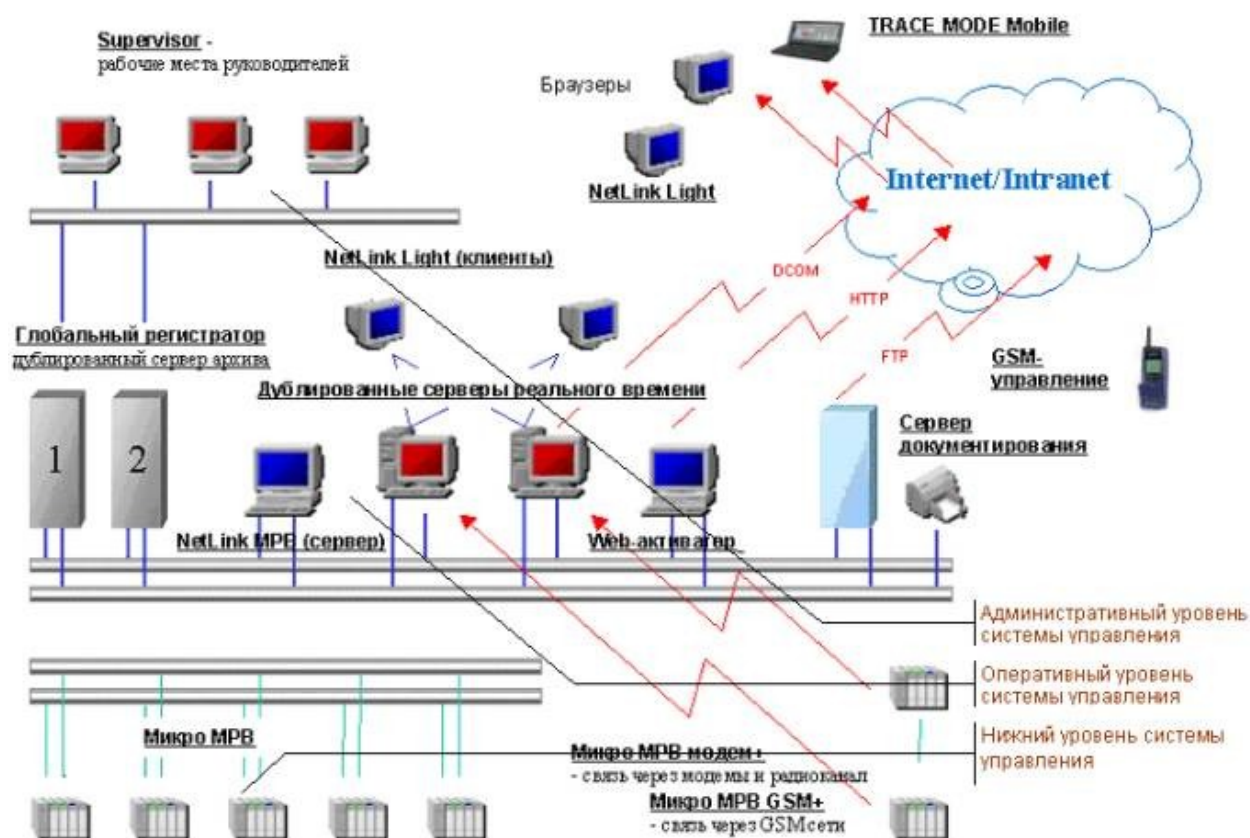
Уровень контроллеров – нижний уровень;

Уровень операторских станций – верхний уровень;

Административный уровень;

Ниже на рисунке схематично показано разбиение распределенной системы управления по уровням.





Деление на уровни иногда может быть весьма условным. В малых системах функции всех уровней часто реализуются на одной операторской станции. В крупных же на каждом уровне может быть выделена своя иерархия. Тем не менее, в большинстве случаев такое деление правомерно.

Существует ряд программных модулей, назначение которых четко не привязано к функциям одного из перечисленных уровней систем управления. К таким модулям относятся:

- Глобальный регистратор;
- Сервер документирования;
- Web-активатор;
- GSM-активатор.

Они могут использоваться как для создания оперативного, так и административного уровней систем управления.

### Уровень контроллеров

На этом уровне реализуется сбор данных от датчиков и НЦУ. Для создания этого уровня предусмотрены мониторы: **Микро МРВ**, **Микро МРВ Модем+**, **Микро МРВ GSM+**. Первый из них предназначен для запуска в контроллерах, связанных с верхним уровнем по локальной сети или последовательному интерфейсу, второй – при связи по коммутируемым линиям, а третий – по GSM-сети. При использовании выделенных телефонных линий или радиоканалов следует применять первый монитор.

Эти мониторы не имеют графического интерфейса. Однако по математическим функциям они идентичны мониторам верхнего уровня, а также имеют ряд функций, необходимых для работы в контроллерах (например, поддержка сторожевого таймера).

### **Оперативный уровень**

Для верхнего уровня АСУТП предусмотрены такие мониторы, как **MPB**, **NetLink MPB**, **NetLink Light**. Они позволяют создавать рабочие станции оперативного управляющего персонала.

**MPB** может обмениваться данными с другими мониторами ТРЕЙС МОУД, а также с любыми контроллерами через встроенные протоколы или драйвер. Он запрашивает данные у нижнего уровня и передает ему команды управления. Полученные данные могут отображаться, архивироваться и передаваться другим приложениям WINDOWS по протоколам ODBC, OPC и DDE.

**NetLink MPB** – это сетевая рабочая станция. Этот монитор может обмениваться данными по последовательному интерфейсу и локальной сети с Микро MPB, работающими в контроллерах нижнего уровня, и операторскими станциями. По функциям визуализации, архивирования, связи с базами данных и документирования **NetLink MPB** аналогичен **MPB**. Этот монитор не поддерживает обмен с внешним драйвером, а также обмен по встроенным протоколам.

**NetLink Light** – это сетевой графический терминал. Он не имеет своего сервера матобработки, а связывается с сервером **MPB** или **NetLink MPB**, запущенным на другом компьютере. **NetLink Light** позволяет создавать дополнительные рабочие места оператора.

### **Административный уровень**

Задачей данного уровня управления является контроль текущего состояния производственных процессов и анализ функционирования производства по архивным данным.

Для решения задач данного уровня предусмотрен монитор **SUPERVISOR**. Он является специализированной графической консолью, которая может подключаться к серверу матобработки **MPB**, **NetLink MPB** или ГР. В первых двух случаях просматривается локальный СПАД архив, а в последнем – глобальный архив. Кроме того, **SUPERVISOR** можно переключить в режим реального времени. В этом случае он работает как консоль **NetLink Light**, и может использоваться для управления процессом.

При работе с архивами **SUPERVISOR** реализует следующие функции:

- отображение последних изменений значений каналов;
- просмотр архивов в режиме **PLAYBACK**;
- просмотр на заданное архивное время с пошаговым переходом по времени.

### ***Линии передачи данных***

#### [Последовательный интерфейс](#)

## Локальная сеть

## Радиоканал

## Коммутируемые линии

## GSM сети

Мониторы реального времени ТРЕЙС МОУД могут обмениваться данными по следующим линиям:

локальная сеть;

последовательный интерфейс RS-232, RS-485, RS-422;

радиоканал;

выделенная телефонная линия;

коммутируемые телефонные линии;

сети GSM.

По этим носителям организовать информационные потоки всех уровней системы управления. При этом могут реализоваться как вертикальные связи (между уровнями), так и горизонтальные (между узлами одного уровня).

### **Последовательный интерфейс**

Обмен по всем линиям, кроме локальной сети, реализуется через последовательный порт по протоколу M-LINK. Для перехода на любую из них нужен соответствующий конвертер или модем.

Узлы в сети M-LINK неравноправны: один имеет статус MASTER, а остальные – SLAVE. Такие сети следует использовать для связи между операторскими станциями и контроллерами.

### **Локальная сеть**

Для построения локальной сети могут использоваться различные адаптеры: ETHERNET, ARCNET, TOKEN RING и др. Существенно, чтобы для них в сетевой операционной системе имелись соответствующие драйверы. Объединенные в локальную сеть операторские станции образуют оперативный уровень управления. На нем могут создаваться рабочие дублирующие друг друга места. Для этого можно использовать различные комбинации мониторов ТРЕЙС МОУД.

Первый вариант – это использование дублированных **МРВ**. Для них постоянно поддерживается контроль работоспособности. В случае необходимости осуществляется их автоматическое переключение с горячего резерва на рабочее состояние.

Второй вариант тиражирования рабочих мест предполагает использование на одной операторской станции **МРВ**, а на остальных – **NetLink Light**. Функции связи с

контроллерами и архивированию выполняет **MPB**. Остальные станции являются графическими терминалами. Они связываются по сети с базой каналов **MPB**.

Для связи оперативного уровня с административным необходимо использовать локальную сеть. В этом случае **SUPRVISOR** запрашивает данные из архивов на удаленном диске.

### **Радиоканал**

Если автоматизируемый объект распределен на большой территории и нет возможности использовать локальную сеть следует применять сети на базе протокола **M-LINK**. В этих случаях можно использовать радиоканал, коммутируемые линии или **GSM** сеть.

Для использования радиоканала к последовательным портам компьютеров следует подключить радиомодемы. Они преобразуют электрические характеристики порта **RS-232** в сигналы, передаваемые на радиостанцию для посылки в эфир, и выполняют обратное преобразование. Можно использовать любые марки радиомодемов, обеспечивающие прозрачность передачи данных между компьютерами и не требующие команд управления от компьютера.

Для обмена по радиоканалу не нужны дополнительные драйверы. Он настраивается так же, как обмен с удаленными узлами по последовательному интерфейсу.

### **Коммутируемые линии**

Для обмена данными между узлами **ТРЕЙС МОУД** при автоматизации сильно распределенного по территории объекта можно воспользоваться коммутируемыми телефонными линиями. Для обмена используется протокол **M-LINK**.

Этот обмен предполагает наличие у каждого узла своего телефонного номера (в некоторых случаях может потребоваться два номера). При организации такой связи на верхнем уровне используется **MPB Модем+**, а на нижнем – **Микро MPB Модем+**.

### **GSM сети**

Этот тип линий передачи данных предполагает наличие сотовой связи в регионе, где находится автоматизируемый объект. Для обмена по **GSM** сети к последовательным портам компьютеров должны быть подключены **GSM-модемы**.

Обмен данными между мониторами **ТРЕЙС МОУД** по **GSM** сети реализуется в виде **SMS-сообщений (Short Message Service)**. Для поддержки такого обмена на уровне операторских станций предназначен исполнительный модуль **GSM-активатор**, а на уровне контроллеров – специализированная модификация микро монитора реального времени **Микро MPB GSM+**.

**GSM-активатор** устанавливается на том же компьютере, где инсталлирован **MPB** или **NetLink MPB**, которые должны обмениваться по **GSM** сети. Он обеспечивает поддержку обмена данными в виде **SMS-сообщений** с другими **MPB** и **Микро MPB GSM+**. Кроме того, он обеспечивает отсылку на сотовые телефоны аварийных или технологических сообщений, а так же контроль и управление технологическими параметрами с сотовых телефонов.

## Сетевой обмен

ТРЕЙС МОУД позволяет создавать крупные распределенные системы. В одном проекте может присутствовать до 128 сетевых серверов. Это могут быть контроллеры, работающие под управлением **Микро МРВ**, и операторские станции, работающие под управлением **МРВ** и **NetLink МРВ**. В той же сети могут присутствовать дублированные глобальные регистраторы, а так же неограниченное число архивных станций, работающих под управлением **SUPERVISOR**, и консольных операторских станций, работающих под управлением **NetLink Light**.

Все серверы реального времени имеют уникальный сетевой номер. Этот номер присваивается узла при его добавлении в проект.

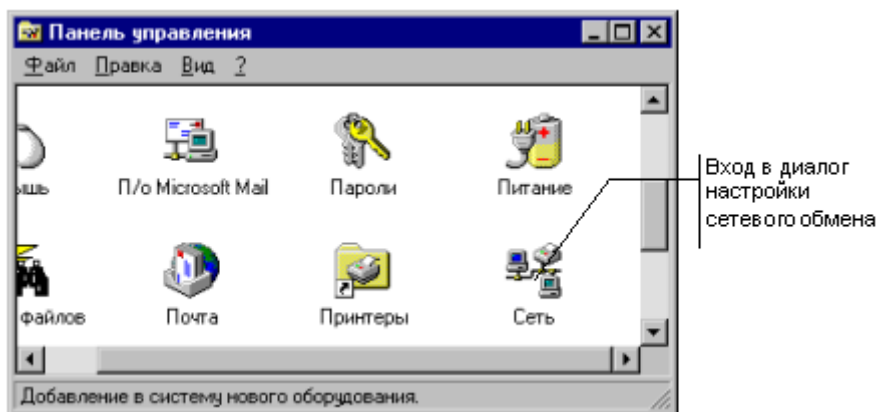
**Внимание!** Сетевой номер задается в диапазоне от 1 до 199. Он является идентификатором узла в рамках проекта и не связан с идентификаторами сетевой ОС.

Глобальные регистраторы всегда имеют сетевой номер 200. Что касается **SUPERVISOR** и **NetLink Light**, то они являются клиентами. Поэтому их число в сети теоретически не ограничено.

### Используемые сетевые операционные системы

Для обмена между мониторами ТРЕЙС МОУД по сети рекомендуется использовать сети WINDOWS. Чтобы подключить к ней узлы, работающие в среде MS DOS, нужен соответствующий клиент WINDOWS или сетевая компонента от WINDOWS 3.11.

Для обмена данными по сети между мониторами ТРЕЙС МОУД необходимо настроить и запустить сетевую операционную систему. В WINDOWS для настройки сети надо дважды нажать ЛК на соответствующем ярлыке панели управления.



Минимальная конфигурация для обмена данными между мониторами ТРЕЙС МОУД, включает в себя:

- службу **Клиент для сетей Microsoft**;
- драйвер сетевой платы;
- сетевой протокол **NetBEUI**.

### Связь с мониторами версии 4.20

Связь по сети с мониторами версии 4.20 осуществляется только в одном направлении - прием данных, передаваемых по автопосылке.

Для реализации такой связи необходимо вставить базу каналов узла, работающего под управлением МРВ 4.20, в структуру проекта. Это осуществляется командой **Импорт версии 4.23x** из меню **Узлы**. При этом выбранная база каналов конвертируется в новый формат, а все каналы с флагами автопосылки добавляются в список сетевых посылок. Значения каналов, присутствующих в этом списке, доступны для приема на любом узле проекта.

При ручном прописывании каналов связи с узлом МРВ 4.20 (без импорта базы этого узла) связь работать не будет, т.к. в файле **addr.ind** не будет соответствующей строки описания этого узла.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 12. [Разработка распределённых систем Trace Mode 5](#). Справочная система Trace Mode 5, раздел: «Разработка распределённых систем», главы: «Обзор раздела», «Идеология распределённых комплексов», «Сетевой обмен по протоколу NetBEUI», «Обмен по протоколу M-Link.**

ЛЕКЦИЯ 13  
**INTERNET/INTRANET-РЕШЕНИЯ И SCADA-СИСТЕМЫ.  
СТРАТЕГИЯ КЛИЕНТСКИХ ПРИЛОЖЕНИЙ**

1. Структура Windows DNA;
2. Новая реализация клиентского приложения в режиме сервер/терминал;
3. Стратегия клиентских приложений от Wonderware;
4. Internet/Intranet решения от CiTechnologies;
5. Общие тенденции и различие реализаций;
6. Управление через Интернет в Trace Mode 5.
  - консоль управления Web-активатора;
  - создание Web-страниц на сервере;
  - доступ к проекту через Интернет.

Тема обеспечения доступности данных производственного технологического процесса с любого компьютера предприятия, с любой подсистемы стала актуальной. И SCADA-приложения должны быть источником технологических данных, с одной стороны, и их потребителем, с другой. Кроме того, различного типа клиентские приложения могут предоставлять соответствующие производственному процессу в огромном объеме данные в приемлемом для пользователя виде. Основная идея данной главы - рассмотреть типы клиентских приложений и протоколы, используемые для передачи, как исторических данных, так и данных реального времени.

Самым простым и распространенным клиентским приложением являются клиенты в локальной сети (рис.1).

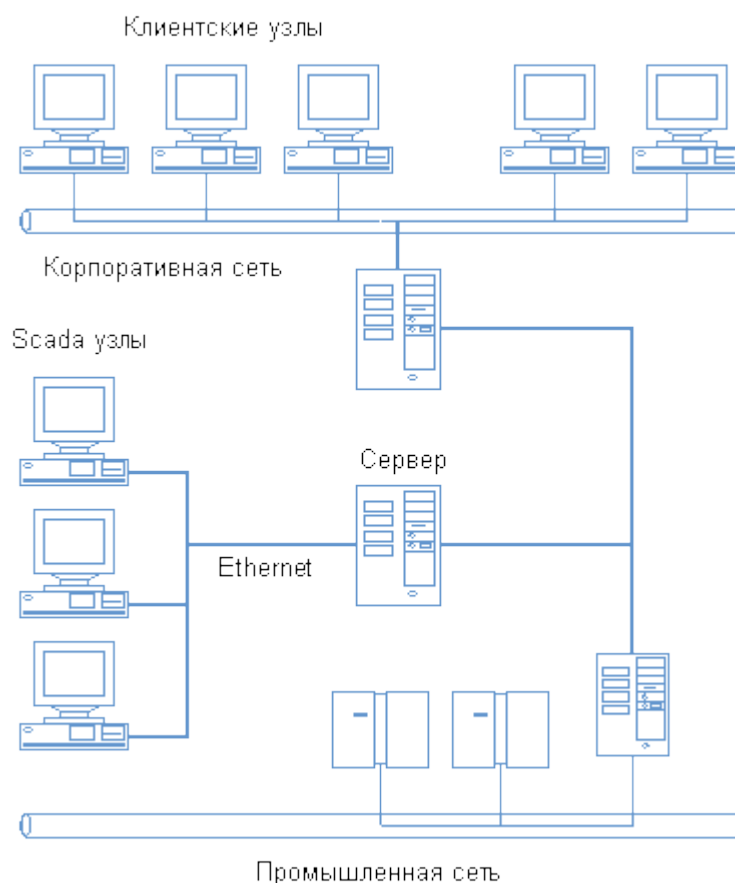


Рис. 7.1. Традиционное решение.

Клиент-серверная организация SCADA-систем предполагает применение наряду с серверными конфигурациями клиентских компонент двух типов: с возможностью передачи управляющих воздействий с клиентского приложения и чисто мониторинговые приложения. Такие клиентские компоненты SCADA-системах традиционно объединяются с серверными приложениями протоколами локальных сетей (TCP/IP, NetBEUI). Но Internet/Intranet технологии не оставили безучастными разработчиков SCADA-систем, баз данных реального времени и др. и привели к появлению следующих типов клиентских приложений:

- клиентские приложения в режиме сервер/терминал;
- бедные и богатые Internet/Intranet-клиенты

Основой рассматриваемых решений для клиентских приложений являются новые технологии Microsoft, реализованные в структуре Windows DNA (Distributed Internet Architecture). Поэтому предлагается начать изложение с краткого изложения особенностей этой структуры.

В второй части статьи рассматривается специальный инструментарий для создания Internet/Intranet – клиентов.

## 1 Структура Windows DNA

Структура Windows DNA - это, в первую очередь, реализация трехуровневой модели приложения, включающей (рис.2):

- уровень представления;
- уровень бизнес-логики;
- уровень доступа к данным.



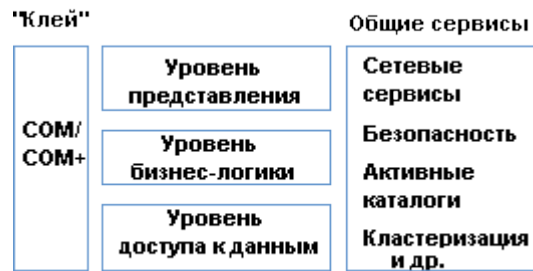


Рис. 7.2. Структура Windows DNA.

Кроме технологий, "привязанных" к уровням, применяются и технологии, представляющие общие сервисы и "склеивающие" технологии. В программном обеспечении Microsoft роль "склеивающих" технологий играют COM и COM+. COM (Component Object Model, архитектура компонентных объектов) - это объектно-ориентированная технология. С компонентной организацией приложение конструируется из COM-объектов, используя готовые наборы этих объектов.

**Слой Windows DNA.** Технологии Microsoft и относящийся к ним инструментарий предназначены для разработки и реализации трехуровневых приложений.

Уровень представления. Есть два обширных вида клиентов, называемых "бедным" (thin) и "богатым" (rich) клиентом. В отличие от толстого (fat) богатый клиент в большей степени ссылается на используемые при создании пользовательского интерфейса технологии, чем на то, какое количество кода выполняется на стороне клиента. Богатые клиента похожи на обычные приложения Win32, но они представляют собой клиентскую часть трехуровневого приложения.

Бедные клиенты не одинаково бедны. Примером бедного клиента служит давно известный терминал. Microsoft предложил технологию Windows Terminal Server, в которой приложение Windows работает на центральном сервере и передает графический интерфейс пользователю-клиенту. Но при этом требуется дорогостоящий сервер, широкая полоса пропускания между клиентом и сервером. Но чаще всего понятие бедный клиент обозначает приложение, работающее на Web-сервере и передающее пользовательский интерфейс с помощью HTML-страниц на Web-браузер.

Далее появилась идея обогащения Web-приложений различными компонентами, которые могут использоваться браузером, - управляющие элементы ActiveX, апплеты Java и т.д. Различной оснащенности бедные клиенты предлагаются и компаниями-поставщиками SCADA-систем.

Уровень бизнес-логики. Три сервиса свойственны этому уровню: сервисы компонентов (COM), Microsoft Message Queue (MSMQ) и Internet Information Server (IIS). IIS - полнофункциональный Web-сервер Microsoft, интегрированный в Windows 2000 Server. IIS является сервером приложений, поддерживающим бедных клиентов, которые подключаются к нему через протокол HTTP.

Microsoft Transaction Server и COM+. Транзакция является фундаментальной структурной концепцией, которая обеспечивает разработку сложных многопользовательских приложений для работы с данными. Главное свойство транзакции в атомарности. Именно концепция транзакции обеспечивает выполнение ряда операций получения данных из разных СУБД и позволяет рассматривать их как единую операцию (рис.3).

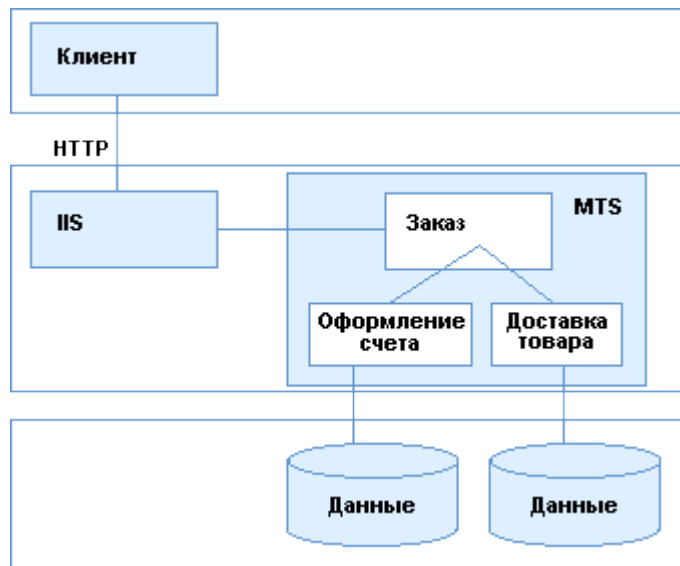


Рис. 7.3. Трехуровневое приложение.

Microsoft Message Queue - асинхронная однонаправленная связь, ориентированная на сообщения. Как DCOM, так и HTTP - синхронные протоколы, которые возвращают результат, до получения ответа от сервера работа клиента блокируется. В случае асинхронного MSMQ вызов сервиса осуществляется помещением сообщения в очередь. При этом возврат клиенту происходит немедленно (и возврат свидетельствует о постановке в очередь) и клиент продолжает работать (нет блокировки). Уровень доступа к данным. Фундаментальной технологией доступа к данным является OLE DB - гибкий низкоуровневый интерфейс COM.

Структура Windows DNA, особенно уровня представления данных, является основой клиентских приложений, предлагаемых поставщиками SCADA-систем.

## 2 Новая реализация клиентского приложения в режиме сервер/терминал

С появлением Windows NT/2000 Terminal Services вновь стала доступной организация клиентских сессий, когда каждый клиент функционирует независимо от других. В этом случае каждый пользователь получает свой ресурс: память, время центрального процессора, доступ к дискам сервера и приложениям. Когда клиент запускается, терминальный сервер регистрирует его, предоставляя доступ к ресурсам сервера. Windows создает также виртуальный дисплей. Затем он передается клиенту и отображается на локальном мониторе. Операции ввода, активизируемые клиентом с клавиатуры, мыши также обслуживаются сервером. Добавление новых клиентов сводится к встраиванию нового терминала.

Для организации взаимодействия между сервером и клиентом используются стандартные протоколы Microsoft RDP (Remote Desktop Protocol) и Citrix ICA (Independent Computing Architecture), что допускает реализацию клиентов в виде супер-тонких бездисковых рабочих станций на платформах Linux/CE, от Windows 3.11.95.98 до рабочих станций Windows NT или 2000.

Используя новые архитектурные возможности, компания-разработчики SCADA-систем имеют возможность предложить терминальные сервисы, поддерживающие выполнение SCADA-приложений в режиме сессии. Так компания Wonderware уже поставляет Terminal Services (терминальные сервисы) для SCADA-системы InTouch версии 7.1, что позволяет установить исполняющую систему InTouch один раз на центральном сервере и затем запускать InTouch-приложения много раз. Клиентские узлы необходимо подключать в режиме терминальной сессии InTouch. Бедный клиент может

быть в этом случае терминалом персонального компьютера или встроенным терминальным устройством с вышеперечисленными операционными системами (рис.4).

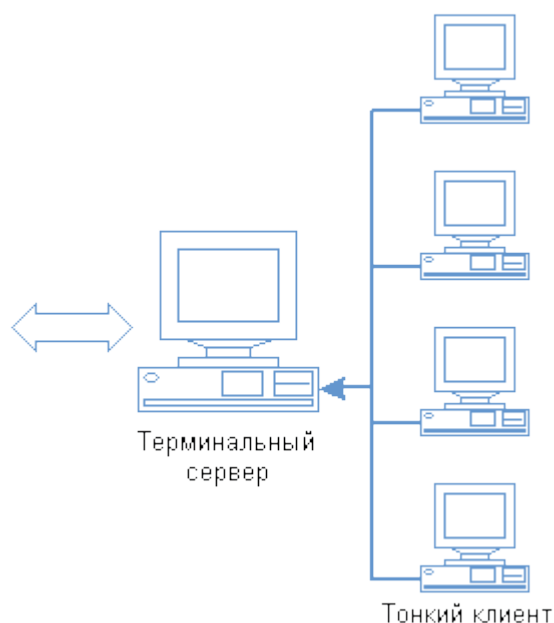


Рис. 7.4. Архитектура терминал-сервер.

Терминальные пользователи имеют доступ к данным, графическим мнемосхемам с возможностью обмена информации в реальном времени без необходимости установки InTouch на локальном клиентском компьютере.

Применение терминал/серверного модели позволяет создавать более экономичные решения за счет того, что приложение устанавливается и поддерживается инженерами только на сервере, использовать клиентские узлы на различных платформах. Следует заметить, что на клиентских узлах может просматривать как одно и тоже приложение, так и разные приложения.

### 3 Стратегия клиентских приложений от Wonderware

#### 3.1 Бедные и богатые Internet/Intranet-клиенты

В Internet/Intranet решениях в обмене данными, кроме технологического сервера, как поставщика данных, и клиента, как получателя информации, задействован Web-сервер (рис. 5). Информация на сервере хранится в виде страниц, на которых кроме текста могут находиться разные объекты: графические изображения, аудио- и видео ролики, формы для ввода данных, интерактивные приложения и т.д.



Рис. 7.5. Клиенты и серверы Web.

Страницы сервера WWW могут содержать не только статическую, но и динамическую информацию. Страница может содержать формы для выполнения запросов к базе данных. Результат такого запроса будет динамически сформирован в виде страницы, которая появится на экране пользователя. Сервер WWW может решать любую задачу, принимая любые данные от удаленного пользователя, обрабатывая их и передавая обратно. Для обработки на сервере WWW запросов, поступающих от клиентских приложений SCADA и требующих получения данных из БДРВ или других источников информации РВ, разрабатывается специальное серверное расширение, которое с одной стороны получает и обрабатывает динамические запросы от Web-клиентов, а с другой обеспечивает взаимодействие с Microsoft Internet серверами. Взаимодействие между Web-сервером и клиентами осуществляется на основе протокола HTTP (HyperText Transfer Protocol, протокол передачи гипертекста.). Так компанией Wonderware предлагается FactorySuite (FS) Web сервер, который обеспечивает динамическими данными клиента Web, реализованного в виде SCADA-приложения InTouch (врезка 1).

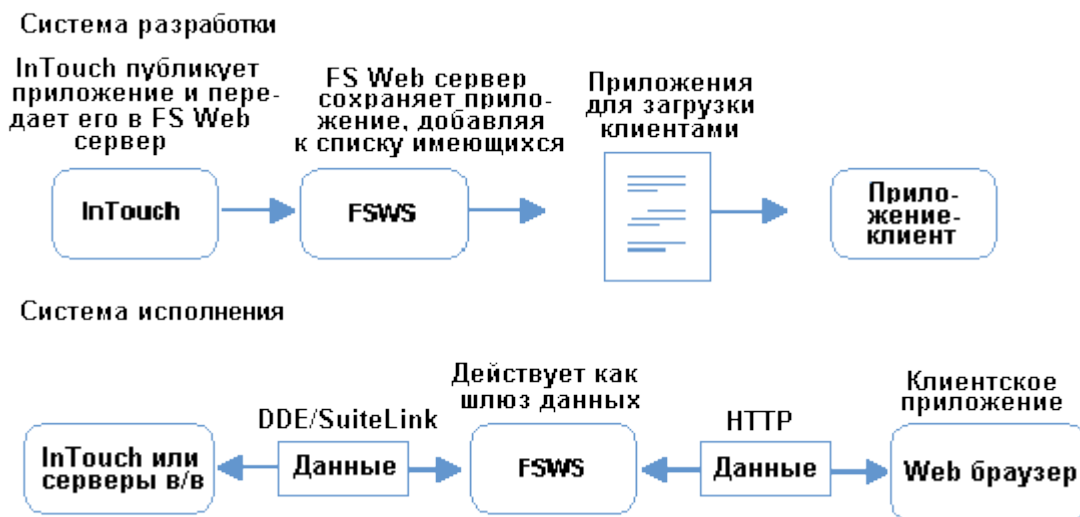


Рис. 7.6. Web сервер для обмена данными между приложениями InTouch.

На рисунке 6 показаны возможности разработки Internet-приложений и запуск их в реальном времени на примере SCADA-системы InTouch.

Причем следует отметить, что процедура публикации (publishing) SCADA-приложений является дружественной и не требует специальной подготовки (врезка 2).

Навигатор Microsoft Internet Explorer (MIE) или исполняющая система InTouch могут использоваться для просмотра приложения web-клиентом. Интернет приложение

позволяет собирать данные с многих FS Web серверов (рис.7). В таких случаях каждый Web Server адресуется специально именем или IP-адресом. Чтобы подписаться на приложение необходимо загрузить его из текущего FS Web сервера и выделить его в локальную директорию на клиентской машине.

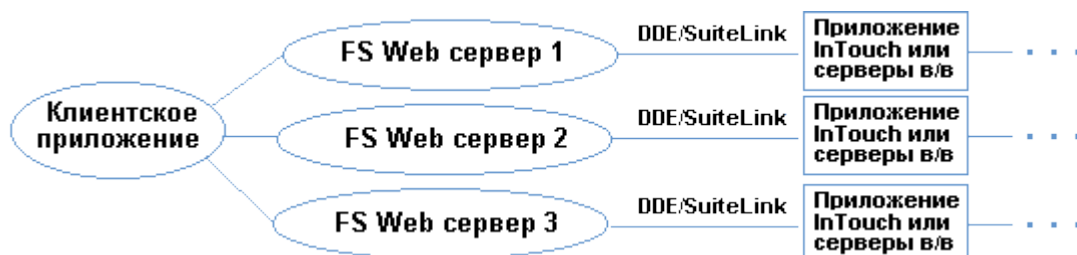


Рис. 7.7. Получение данных от нескольких Web-серверов.

Публикация InTouch приложения возможна в двух режимах: с исходными файлами, так что приложение может модифицироваться в среде разработке в дальнейшем и в режиме исполнения только.

Таким образом, приложения некоторых SCADA-систем могут поддерживать функцию толстого или богатого Internet-клиента. Преимущество применения такого клиента в том, что способ разработки клиентского приложения остается традиционным (обычное SCADA-приложение), возможно использование режима управления. А недостатком, безусловно, является то, что для каждого клиентского узла оплачивается лицензия.

Если клиент является бедным, то обработка любого запроса клиентского приложения выполняется на сервере. Только Web-страница предоставляется клиенту. Рассмотрение такого типа клиентов начнем с клиентов к базам данных (БД).

### 3.2 Базы данных реального времени (БДРВ) и Internet-решения

Поскольку БДРВ поддерживают язык SQL-запросов, то для организации доступа к технологической информации возможен стандартный подход как к обычным реляционным БД. Традиционный подход позволяет получать данные из БД и БДРВ, используя уже ставшие стандартными SQL-объекты, доступные, практически, из любого броузера. Этот подход требует программистского опыта разработки web-сайтов и использования специальных SQL- объектов, но является типичным примером бедного клиента.

Рассмотрим сейчас более простую, с точки зрения пользователя-разработчика сайта, процедуру доступа к БДРВ на примере IndustrialSQL Server от Wonderware. IndustrialSQL Server использует трехуровневую клиент-серверную архитектуру (рис. 8), которая позволяет создавать Интернет/интранет приложения. Обработка запроса на получение данных, сделанного клиентским объектом к IndustrialSQL Server, поддерживается с помощью специальных объектов Business Objects. Специальные объекты являются COM (Component Object Model) объектами, которые размещается либо на локальном компьютере либо на Microsoft Internet Information Server (IIS) и в этом случае он доступен через интернет и отвечает за получение данных из БДРВ.

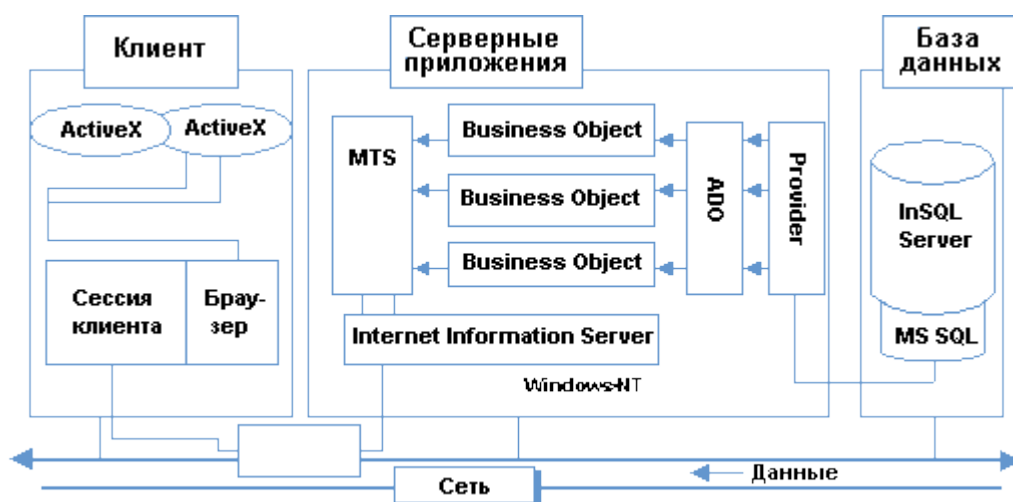


Рис. 7.8. Трехуровневая клиент-серверная архитектура.

**Клиентские приложения.** Формат таблиц базы данных в БДРВ, в основном, предопределен. И клиентские приложения, учитывая предопределенный формат таблиц, обеспечивают доступ к данным для визуализации и анализа. Клиентские приложения не требуют от пользователя знания языка SQL-запросов, что расширяет класс пользователей. Так для Plant2SQL (CiTechnologies), так для IndustrialSQL Server компании предлагают специальные приложения, ориентированные на получение данных из БДРВ. С технологической точки зрения часть приложений реализованы как независимые приложения, другая часть представляет ActiveX объекты.

Как независимые приложения, встроенные в программы Microsoft Office, так и ActiveX объекты предназначены для создания текущих и архивных трендов, для создания параметрических графиков X-Y и для табличного отображения текущих и архивных данных. На врезке 3 описаны ActiveX-объекты из пакета FactoryOffice компании Wonderware.

ActiveX объекты могут встраиваться в приложения InTouch, Visual Basic, Visual C, и в HTML-страницы Internet Explorer. А специальные серверные компоненты - Business Objects - обеспечат получение данных запрошенных в ActiveX объекте или SQL-запросе. Использование ActiveX-технологии, с точки зрения клиентских приложений, сводится к настройке на интернет-обмен при конфигурировании соответствующего ActiveX-объекта: для этого активизируется свойство Use Internet Server (Использовать Internet сервер) и определяется имени или IP-адрес сервера в форме HTTP:// имя сервера.

Использование ActiveX-объектов оснащает бедных клиентов новыми возможностями, т.е. бедные клиенты не одинаково бедны.

### 3.3 Специальный инструментарий для создания Internet/Intranet – клиентов

Если Вы не используете готовых приложений - клиентов Web, то для того чтобы создать свой Web-сайт и при этом разрабатывать не просто бедного клиента, а оснащенного ActiveX-объектами, Java-апплетами и др. целесообразно рассмотреть используемый для этого инструментарий. Инструментарий является разноуровневым: традиционный инструментарий общего назначения и ориентированный на особенности механизмов обмена, используемых в АСУТП. Специализированный инструментарий характеризуется тем, что поставляют его сейчас:

- независимые компании (Intuitive Technology), предлагающие поддержку характерных для АСУТП протоколов (DDE, OPC, OLE DB), таким образом, обеспечивая клиентские приложения и данными в реальном времени;

- компании-разработчики SCADA-систем. Их инструментарий поддерживает не только ставшие стандартными протоколы обмена, но частнофирменные протоколы, конвертацию приложений SCADA в HTML, XML-языки. Как пример такого инструментария рассмотрим SuiteVoyager от Wonderware.

Создания собственного или редактирование существующего web-сайта. Пользователь устанавливает соединение с сервером WWW через сеть с помощью специальной программы просмотра страниц WWW - браузера, например, навигатор MIE и Netscape Navigator. При установке соединения пользователь указывает адрес сервера WWW. Дополнительно он может указать путь к файлу страницы WWW, которая должна быть отображена сразу после подключения к серверу. К серверу может подключаться несколько Web-сайтов. Web-сайт - это не просто набор отдельных Web-страниц, а иерархическая система HTML документов, файлов, графических изображений, апплетов на языке Java, текстовых видео- и аудио файлов, а также сценариев на CGI или ином языке. Для обеспечения целостности сайта используются гипертекстовые связи (hyperlink)- врезка 4.

Для создания сайтов предлагается сейчас на рынке разнообразие инструментальных средств и их выбор зависит в первую очередь от решаемых задач. Для создания сайтов, ориентированных на мониторинг и управление технологическим процессом, предлагается использовать пакеты Microsoft InterDev или FrontPage. Рассмотрим особенности последнего. Итак, FrontPage используется как:

- визуальное средство, позволяющее непрограммистам реализовать web-публикацию в среде клиент/сервер.
- FrontPage используется для обслуживания Web-сервера и web-сайтов на этом сервере.
- Web-страница с FrontPage поставляется с 16 и 32-разрядными версиями собственного сервера Personal Web Server, который может использоваться с ОС Windows 3.11, Windows 95, Windows NT.

Программное обеспечение Web-сервера, ответственное за обработку полученных от клиента данных, динамическое формирование HTML документа и возврат его пользователю, должно быть установлено перед установкой пакета FrontPage. Серверные расширения FrontPage поддерживают стандарты HTTP и CGI, обеспечивая совместимость с существующими HTML документами и CGI-сценариями (врезка 5).

Текстовые файлы страниц готовятся с использованием специального языка разметки гипертекста HTML (Hyper Text Markup Language); Взаимодействие пользователя с сервером WWW осуществляется через формы. Сервер, получив данные из полей формы, запустит созданное специально для этой формы программное расширение для обработки полученных данных, динамически сформирует документ HTML и возвратит его пользователю (нет ограничений на вид выполняемой обработки или вид сформированного документа HTML).

Сервер, содержащий наряду со статическими динамические документы, называют активным интернет-клиентом. Активные серверы создаются и использованием программных расширений сервера WWW- приложений CGI, ISAPI (врезка 6).

Данные, полученные через запросную форму, передаются программному расширению CGI или ISAPI. Эти расширения могут обратиться, например, к СУБД через интерфейс ODBC или через интерфейс этой СУБД, а результат запроса оформить в виде документа HTML и вернуть удаленному пользователю.

Возможности языка HTML ограничены. Часто требуется обрабатывать содержимое локальных файлов, отображать данные в графическом виде или выполнять др. нетривиальную работу. Создав орган управления ActiveX и расположив его на сервере WWW, можно сделать ссылку на этот орган в документе HTML.

Код ActiveX загружается из сервера WWW в адресное пространство удаленного компьютера и поэтому имеет доступ ко всем его ресурсам. Это позволяет организовать сложные алгоритмы обработки и отображения любых локальных данных, что невозможно при использовании программных расширений CGI и ISAPI. Но ActiveX представляет и потенциальную угрозу в смысле распространения вирусов. Для уменьшения угрозы MS предложила сертификацию органов управления ActiveX. Когда пользователь попадает на страницу со ссылкой на ActiveX, ему выдается изображение сертификата фирмы разработчика. Если пользователь доверяет сертификату, он может согласиться на загрузку и запуск ActiveX, если нет - можно отказаться.

Язык HTML допускает использование языков программирования Java, JavaScript и VBScript. Язык программирования Java разработан фирмой Sun на основе языка Oak, как платформенно-независимый интерпретируемый, объектно-ориентированный язык. Создаются программы Java и размещаются ссылки на них в документах HTML. Такие Java-программы называются апплетами (applets). Программы Java, расположенные на сервере WWW, обладают большими возможностями по обработке и отображению данных. По сравнению с ActiveX объектами они более безопасны, поскольку не могут выполнять запись на локальные диски и читать с них.

Исходный текст программ, составленных на языке программирования JavaScript и VBScript, вставляется непосредственно в документ HTML, поэтому для их разработки не нужны специальные средства. Интерпретатор JavaScript и VBScript встроен непосредственно в браузер Microsoft Internet Explorer (Netscape не работает с языком VBScript).

Страницы сервера WWW содержат ссылки на другие страницы, реализованные в виде специальных текстовых строк, либо в виде графических объектов или органов управления. Страницы могут ссылаться на страницы, расположенные и на других серверах в сети интернет, включая серверы FTP, Gopher, конференции, электронные почтовые адреса.

Следует сказать о языке XML (Extensible Markup Language), имеющего общего предка с HTML - стандартного обобщенного языка описания документов SGML (Standard Generalized Mark-up Language), но XML имеет более строгий синтаксис. Отмечается тенденция: HTML - язык для представления данных, а не для обмена ими, в то время как публикация данных происходит в формате XML. Производители Oracle, Sybase, Informix скоро начнут выдавать результаты запросов в формате XML и импортировать XML-данные в свои таблицы.

Таким образом, используя инструментальные средства подобные FrontPage, Вы можете создать собственные, ориентированные на решение Ваших задач web-сайты. Предлагаемые технологии Microsoft, позволяют применять как ActiveX технологию, так и технологию доступа к реляционным базам данных. Их использование допускает встраивание:

- рассмотренных ранее ActiveX объектов для доступа к данным IndustrialSQL Server (ActiveTagBrowser, ActiveDataGrid, ActiveGraph, ActiveTimeSelector);
- стандартных форм SQL-запросов ряда браузеров (прежде всего, Microsoft Internet Explorer).

Сервер WWW может решать любую задачу, принимая любые данные от удаленного пользователя, обрабатывая их и передавая обратно.

Пакет SuiteVoyager. Специальный пакет от Wonderware SuiteVoyager предоставляет масштабируемое, расширяемое средство разработки информационных порталов. Портал является просто Web-сайтом, который предоставляет пути доступа к дополнительной информации по определенным темам. SuiteVoyager является набором интегрированных программ, поддерживающих удобный способ для получения технологической информации (рис. 9).



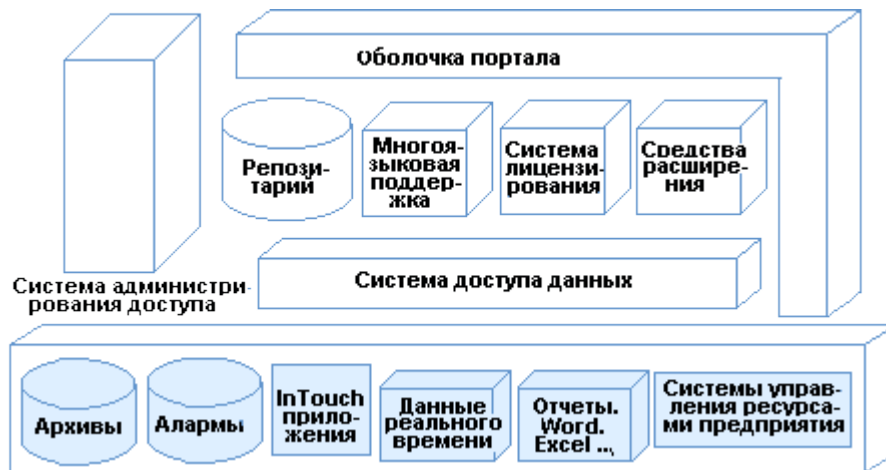


Рис. 7.9. Структура портала SuiteVoyager.

Пакет представляет набор средств для просмотра, подготовки отчетов на основе технологических данных. Традиционно передача графической информации требует доставки файлов большого размера и длительных периодов времени для загрузки. Чтобы преодолеть это ограничение, SuiteVoyager поставляет интерактивные HTML-страницы, преобразуя существующие графические окна InTouch (и ассоциированную с ними анимацию) в XML (рис. 10).

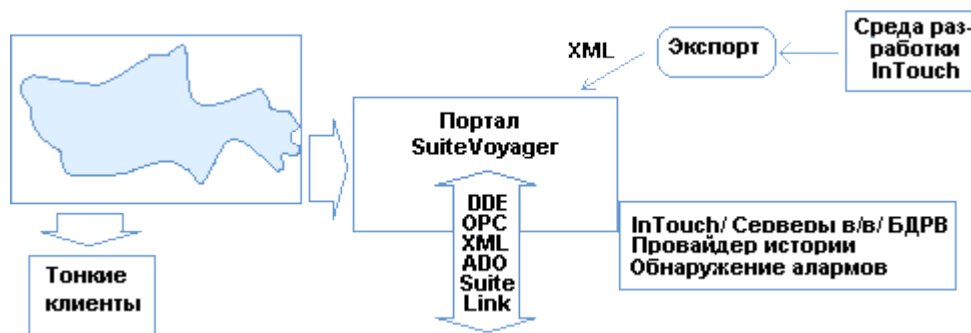


Рис. 7.10. Решение на основе SuiteVoyager.

Использование XML-технологии уменьшает объем передаваемой между клиентом и сервером информации почти на 80%. SuiteVoyager позволяет пользователям визуализировать технологическую информацию, поступающую из серверов ввода-вывода, SCADA-приложений, БДРВ через Internet/Intranet, используя Internet Explorer 5+. Пакет поддерживает новую "made-for-the-Web" технологию, такие как XML (eXtensible Mark-up Language).

#### 4 Internet/Intranet решения от CiTechnologies

Пакет Plant2Business от CiTechnologies - это целое семейство экономически эффективных и удобных в использовании программных средств превращения технологических данных в информацию, доступную каждому работнику организации. Интеграция технологических и административных информационных систем посредством Plant2Business обеспечивает повышение качества принимаемых решений, что в конечном счете благоприятно сказывается на производительности и эффективности работы предприятия.

В семейство Plant2Business входят следующие программные средства:

- база данных Plant2Business Server,
- Web-серверное расширение Plant2NET,
- инструментарий для обмена по GSM - каналам - Plant2Pocket.

Благодаря открытым, стандартным технологиям Plant2Business разрушает стену, традиционно разделявшую технологическую и управленческую информацию. Plant2Business обеспечивает каждому подразделению организации свободный доступ к технологическим данным, предлагая уже знакомые пользователям средства и возможности.

Самая свежая информация становится мгновенно доступной технологам, работникам отделов контроля качества, службам техобслуживания, сбыта и даже клиентам благодаря наличию множества разнообразных средств представления данных. Plant2Business позволяет связывать воедино все и всех - от цеховой площадки до удаленных клиентов в Internet. И все это возможно без какого-либо нарушения ежедневного распорядка работы предприятия.

Применение готовых средств конфигурирования сокращает сроки получения технологической информации с нескольких дней до нескольких минут.

Базой концентрации технологической информации является сервер Plant2Business. Именно к нему могут подключаться различные технологические системы. Соединение с приложением Citect не требует наличия знаний о нем, поскольку сервер Plant2Business автоматически импортирует переменные (Tags), графики (Trends) и тревоги (Alarms), после чего они тут же могут быть опубликованы. SCADA-приложения, такие как Fix(Intellution), InTouch (Wonderware) и др. подключаются через специальные "коннекторы". По двунаправленной линии связи данные могут быть как считаны из, так и переданы в систему управления.

Plant2Net обеспечивает передачу данных из Plant2Business сервера Internet/Intranet клиентам по технологии тонкого клиента. Причем выбираются только необходимые в данный момент данные в виде имеющей смысл иерархической структуры.

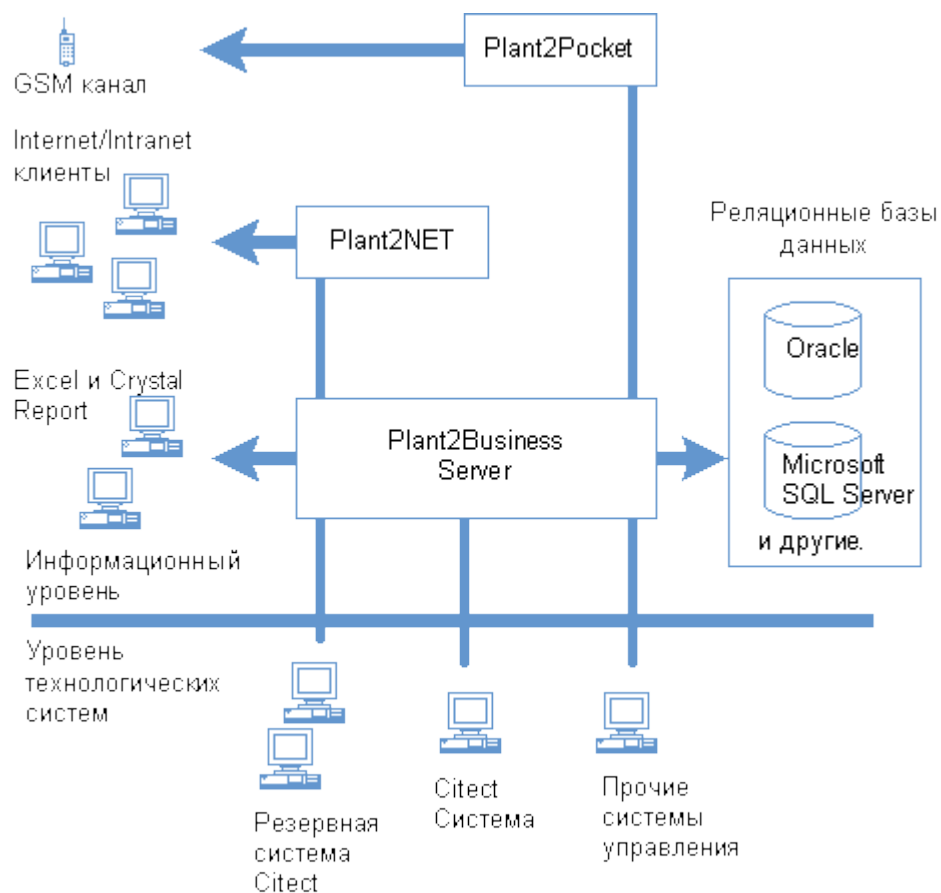


Рис. 7.11. Схема информационных потоков

На рис.11 показана схема информационных потоков: с уровня технологических систем данные поступают в Plant2Business сервер. Клиентские приложения (Excel, Crystal Report, Internet Explorer и т.д.) по различным протоколам обмена имеют доступ к сохраненной в Plant2Business-сервере информации.

Используя предлагаемый пакет обеспечивается возможность осуществлять сбор в реальном времени данных, аварийных сообщений, архивов с различных подсистем и доступ клиентских приложений к ним, в том числе по протоколам http/https.

FTP-клиенты. CiTechnologies поставляет решение обмен данными между приложением Citect, выполняющим функции сервера и клиентскими приложениями (толстые клиенты) по протоколу Ftp.

## 5. Общие тенденции и различие реализаций

Основное назначение клиентских приложений - обеспечить поставку технологической информации из SCADA-систем, баз данных реального времени, или серверов ввода-вывода.

Типичная реализация толстого или богатого клиента часто связана с расширением числа протоколов, которые поддерживают приложения SCADA. С точки зрения пользователя необходимо просто приобретение лицензии исполняющей системы и использование приложения SCADA как Internet/Intranet-клиента.

Два типа бедных клиентов - терминал/серверные и Internet-клиенты могут применяться, хотя последние являются более распространенными. Для организации динамического обмена данными на Web-сервере устанавливаются специальные компоненты, обеспечивающие обмен данными по каналам реального времени (DDE, OPC

и др.) с источниками информации с одной стороны и обслуживающие запросы Web-клиентов по протоколу HTTP с другой стороны.

Web-клиенты способны получать информацию из различных подсистем предприятия или корпорации, включая различные сегменты локальной сети, ориентированные на управление технологическим процессом, подсистемы административно-хозяйственной деятельности и др., просчитывать вторичные параметры, формировать отчеты.

Очевидна тенденция, что клиентские приложения поддерживают протоколы локальных и Internet/Intranet сетей, минимизируя требования к квалификации пользователя в области Internet/Intranet технологий.

При наличии общих тенденций в развитии типов клиентских приложений очевидно различие в их реализации:

- SuiteVoyager, как Web-серверное расширение, обеспечивает получение информации из различных источников реального времени, базы данных реального времени IndustrialSQL Server и предоставление их Internet-клиентам. CiTechnologies подчеркивает значимость сервера Plant2Business как базы регистрируемой со всех источников данных информации;
- CiTechnologies предлагает как TCP/IP, NetBEUI протоколы для обмена по локальной сети, так и по ftp-протоколу для глобальной сети

## 6. Управление через Интернет

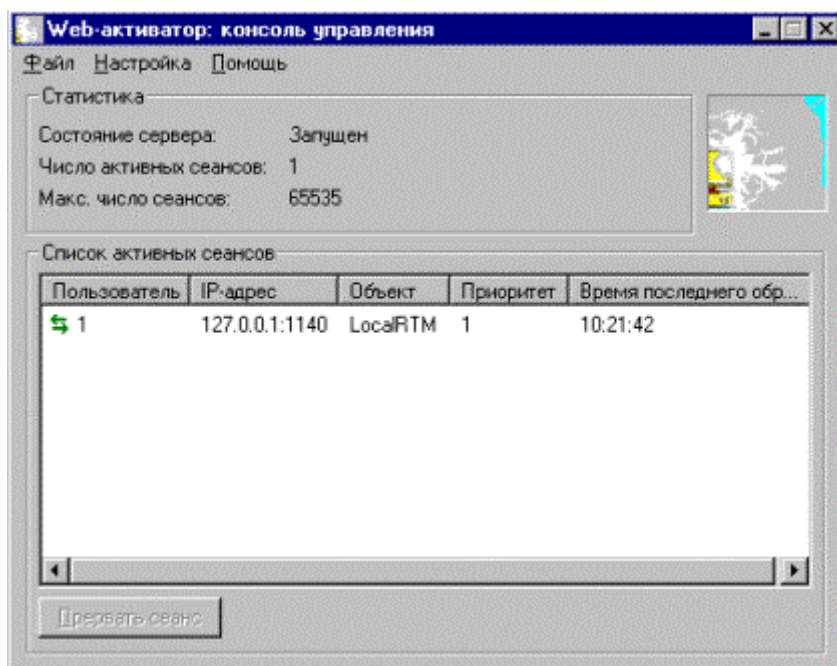
Любая рабочая станция ТРЕЙС МОУД может выступать в качестве Web-сервера, что позволяет управлять технологическим процессом через Интернет. На удаленном компьютере необходимо иметь только доступ к Интернет и Web-браузер. Для реализации данного режима предназначен модуль **Web активатор** исполнительной системы ТРЕЙС МОУД.

### *Консоль управления Web-активатора*



Консоль управления Web-активатора предназначена для установки и конфигурации сервера HTTP. Для ее запуска нужно нажать ЛК на соответствующем ярлыке в программной группе ТРЕЙС МОУД или запустить файл **Web\_Activator.exe**, находящийся в директории инсталляции Web-активатора.

Общий вид консоли управления показан на следующем рисунке:



Окно консоли имеет следующие компоненты:

- главное меню;
- раздел "Статистика";
- рабочая область (список активных сеансов).

### Главное меню консоли управления Web-активатора

[Файл](#)

[Настройка](#)

[Помощь](#)

Главное меню содержит разделы **Файл**, **Настройка** и **Помощь**.

#### **Файл**

Команды меню **Файл** управляют состоянием сервера:

**Управление сервером** – этот раздел включает набор команд по установке, удалению, запуску и остановке сервера HTTP. После старта сервер не требует перезапуска при перезагрузке компьютера;

**Выход** – закрыть окно консоли управления Web-активатора.

#### **Настройка**

Это меню предназначено для настройки сервера. Оно содержит следующие команды:

**Параметры сервера** – задание номера порта сервера HTTP и параметров документирования его работы. Если документирование разрешено, данные сохраняются в текстовом файле с именем **wwwgate\_<MMDD>.log** в директории инсталляции сервера (**MMDD** – месяц и день создания).

**Статические ресурсы** – эта команда открывает окно, в котором созданной Web-странице (ресурсам, размещенным в отдельной директории) присваивается имя (URL).

**Мониторы Trace Mode** – выбор и задание имени (URL) объекта, у которого сервер запрашивает данные.

## Помощь

Данное меню позволяет получить информацию о версии консоли управления Web-активатора и войти в справочную систему.

## Статистика

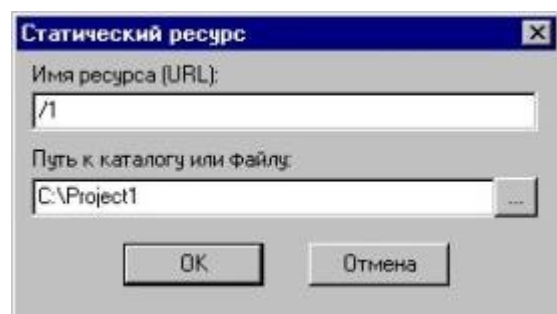
В этом разделе консоли выводятся данные о текущем состоянии сервера и числе активных сеансов (удаленных пользователей, подключенных к серверу). Здесь указывается также максимально доступное количество сеансов.

## Рабочая область консоли

Рабочая область предназначена для отображения атрибутов удаленных пользователей, подключенных в данный момент к серверу. Здесь выводятся имена пользователей, их IP-адреса, приоритет, время последнего обращения к серверу, а также имена объектов, у которых запрашиваются данные. Консоль управления позволяет прервать любой из активных сеансов. Для этого нужно выделить требуемый сеанс и нажать кнопку **Прервать сеанс**, находящуюся в нижней части рабочей области.

## Создание Web-страниц на сервере

Для каждого проекта, к которому удаленные пользователи будут обращаться через Интернет, на сервере создается свой сайт (директория размещения ресурсов). Для этого в меню **Настройка** консоли управления Web-активатора нужно выполнить команду **Статические ресурсы** и в появившемся на экране диалоге **Настройка статических ресурсов** нажать кнопку **Создать**. На экране появится диалог, в котором нужно задать имя создаваемой директории и путь, а также в окне **Имя ресурса (URL)** присвоить этой директории адрес (URL). Имя ресурса должно начинаться со знака / :



При нажатии на кнопку ОК создаются все заданные в диалоге каталоги, в директории ресурсов создается файл **index.htm** (файл с таким именем является по умолчанию первой

страницей сайта) и на экране появляется приглашение открыть этот файл в установленном по умолчанию редакторе Web-страниц. Кроме этого, в директорию ресурсов копируются служебные файлы **graphapplet.jar**, **xp.jar**, **RTM.jar**.

**Внимание!** Путь к директории ресурсов может содержать только латинские буквы.

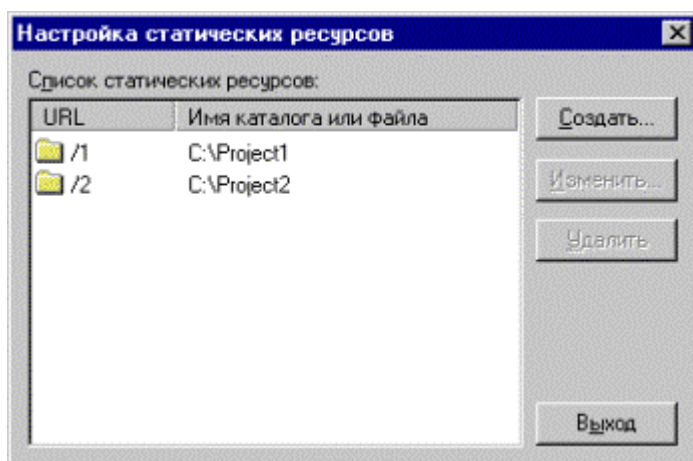
Служебные файлы входят в состав исполнительного модуля **Web\_активатор**. Они выполняют следующие функции:

**graphapplet.jar** – визуализация графических элементов xml-файла;

**xp.jar** – загрузчик xml-файла;

**RTM.jar** – обмен данными с сервером по протоколу HTTP

Примерный вид диалога **Настройка статических ресурсов** после указанной настройки показан на следующем рисунке:



После создания директории ресурсов нужно задать имя объекта для запроса данных. Для этого надо выполнить команду **Мониторы TRACE MODE** меню **Настройка** консоли управления Web-активатора. В диалоге **Мониторы TRACE MODE** следует нажать кнопку **Создать** и ввести имя объекта, а также указать компьютер, на котором он размещен.

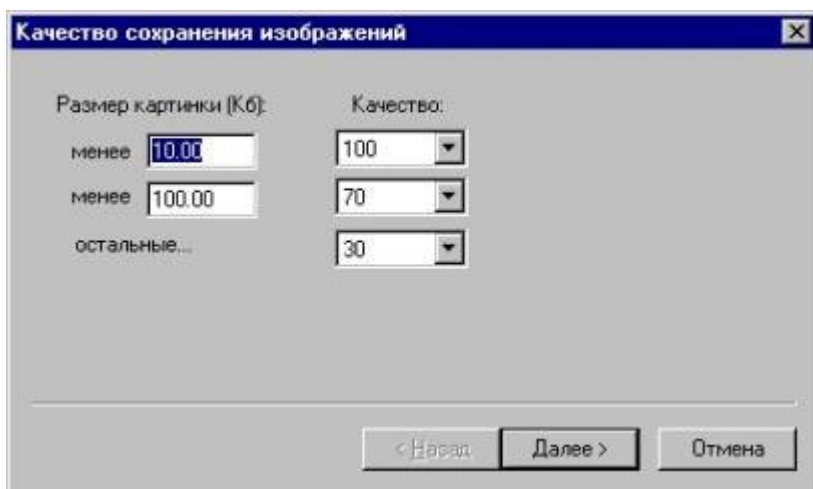
**Внимание!** Для реализации режима управления через Интернет проект должен быть загружен в MPB на указанном компьютере. Кроме того, на компьютерах необходимо произвести настройку механизма DCOM, как указано в разделе [Настройка DCOM](#) (см. **Работа в реальном времени**).

Вид диалога **Мониторы TRACE MODE**: после указанных настроек показан на следующем рисунке:



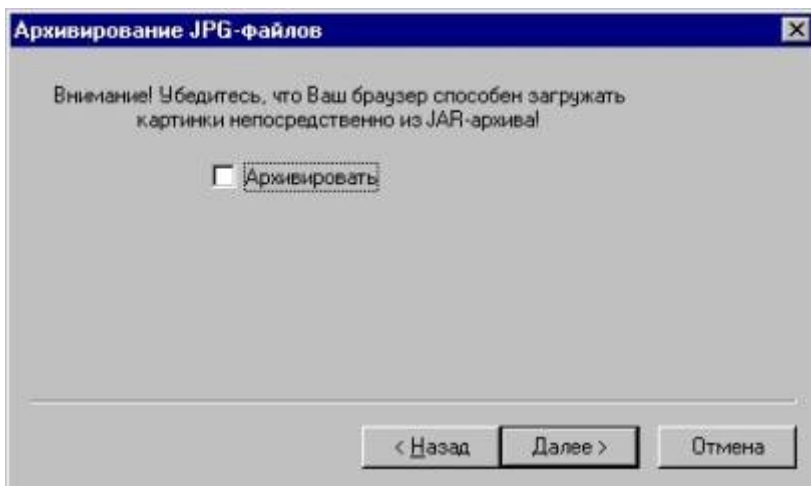
Далее в созданную директорию ресурсов нужно экспортировать из редактора представления данных графическую базу узла в формате **xml** и создать в этой директории HTML-файл, содержащий теги для отображения графической базы в Web-браузере. Для этого нужно загрузить проект в редактор, в бланке **Экраны** навигатора проекта нажать ПК на имени узла, установить курсор на команде **Экспорт** и из открывшегося контекстного меню выполнить команду **Для Web-активатора**. На экране появится диалог сохранения файла в формате **xml**. В этом диалоге нужно задать имя файла и выбрать для его размещения директорию, созданную при настройке статических ресурсов в Web-активаторе.

При нажатии кнопки **Сохранить** на экран выводится следующий диалог:



В этом диалоге в условных единицах от 30 до 100 задается качество сохраняемых **JPEG**-картинок, используемых в графической базе (параметр 100 в разделе **Качество** диалога означает сохранение картинок без потери качества). После установки параметров и нажатия кнопки **Далее** на экране появится следующий диалог:

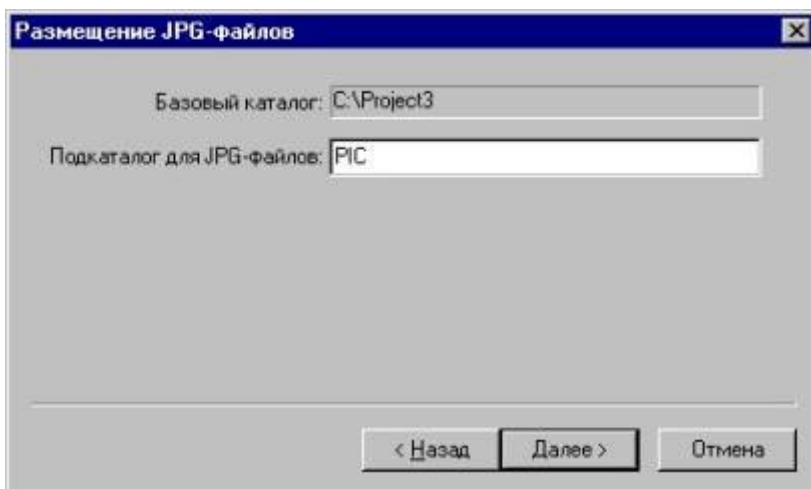




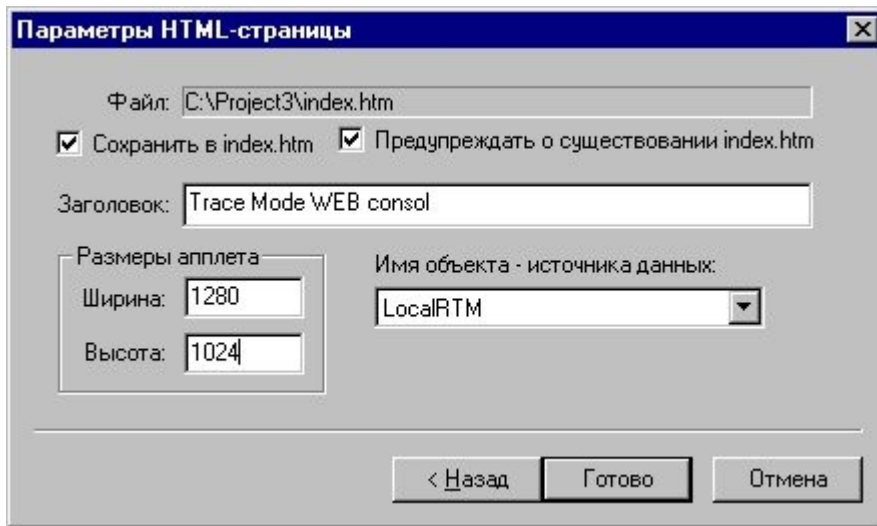
При установке в этом диалоге флага **Архивировать** JPEG-файлы будут сохранены в виде архива JAR. Данный флаг следует установить только в том случае, если Web-браузер позволяет загружать картинки непосредственно из JAR-архива.

С помощью параметров, которые задаются в двух приведенных выше диалогах, можно уменьшить размер ресурсов, передаваемых по сети Интернет удаленному пользователю.

При нажатии кнопки **Далее** на экране появится следующий диалог, в котором можно задать поддиректорию, в которую будут сохранены файлы картинок. Если заданной поддиректории в директории ресурсов нет, она будет создана:



После нажатия кнопки **Далее** на экран выводится диалог задания параметров HTML-страницы, которая будет содержать графическую базу узла:



В этом диалоге можно установить размеры окна (в пикселях), в котором Web-браузер будет отображать графическую базу узла, задать заголовок этого окна, а также выбрать объект-источник данных для создаваемой HTML-страницы (в окне **Имя объекта источника данных** данного диалога отображаются источники, созданные в диалоге **Мониторы TRACE MODE** меню **Настройка** консоли управления Web-активатора).

Создаваемую HTML-страницу можно сохранить как первую страницу сайта, если установить флаг **Сохранить в index.htm**. Если **index.htm** в директории уже есть, при установке второго флага диалога на экране появится соответствующее предупреждение.

Если флаг **Сохранить в index.htm** не установлен, HTML-файл, содержащий теги для отображения графической базы узла в Web-браузере, будет создан с именем, заданным для файла \*.xml. В этом случае **index.htm** должен содержать ссылку на этот HTML-файл.

Ниже приводится пример HTML-файла для отображения графической базы **abt.xml** в Web-браузере:

```
<html>
<head>
<title>Trace Mode WEB visual console </title>
</head>
<body>
<applet archive="graphapplet.jar,xp.jar,RTM.jar" code="GraphApplet" height=1
style="HEIGHT: 480px; WIDTH: 640px" width=1 VIEWASTEXT><PARAM NAME="srcfile"
VALUE="abt.xml">
</APPLET>
</body>
```

</html>

Строка **style="HEIGHT: 480px; WIDTH: 640px"** задает размеры окна, которое Web-браузер откроет для отображения графической базы.

По окончании настроек нужно нажать кнопку **Готово** – файлы \*.xml и \*.htm будут сохранены в директории ресурсов.

### **Функциональные ограничения web-консоли по поддержке элементов графической базы узла**

Общие ограничения:

все линии могут быть только сплошные;

заливки - только сплошные;

формы отображения **Ссылка на экраны** и объекты в окне не имеют заголовка, не могут изменять свое положение и размеры;

не поддерживаются всплывающие подсказки;

не поддерживается наложение окон, кнопок, объектов;

не реализовано вращение графических индикаторов, ФО "Динамический текст", растровых фрагментов, объемных элементов и эллипсов, а также вращение с шагом 90°.

Ограничения по функциям управления:

нет режима действия по нажатию и восстановления значения;

не реализованы послышки по буферу и с добавлением % шкалы;

квитирование - только пустое;

нет ввода комментария и запуска программ.

Нереализованные графические элементы:

метафайлы;

кривые;

гистограммы произвольной формы;

ActiveX (кроме универсального тренда и формы просмотра тревог).

Отсутствующие общие функции:

пропорциональный переход на экран.

Ограничения по элементам:

кнопки: только "мягкие", цвет надписи - системный;

шкалы: не реализовано управление с помощью ползунка; при размещении первым должен быть установлен левый нижний угол;

динамический текст: не поддержан экспоненциальный формат; формат "g" выдается в виде "1 разряд после запятой";

универсальный тренд: нет легенды, on-line редактирования набора кривых, режима движения курсора по узлам, пустые места в архиве не заполняются последними значениями. Не поддерживается перемещение визира на тренде с помощью кнопок => и <=.

### Дополнительные возможности отображения графических элементов

Переопределение цветов в отчете тревог осуществляется заданием дополнительных параметров апплета. Имя такого параметра - ACxy, где

x - номер категории сообщений (0..4):

0 - A,E,W,M,I,R;

1 - !,?,S,Y;

2 - 0,1,2,3,4,5,6,7,8,9;

3 - -,\_,\*;

4 - все прочие;

y - признак: 0 - фон, 1 - текст.

Значение задается шестнадцатеричной константой в формате RGB, как и в XML-файле. Например,

```
<PARAM NAME="AC31" VALUE="#F00000">
```

```
<PARAM NAME="AC30" VALUE="#0000F0">
```

задают цвет красный по синему для категории сообщений 3.

Для апплета могут быть также заданы следующие параметры:

**exch\_period** – период опроса сервера в миллисекундах, значение по умолчанию – 500;

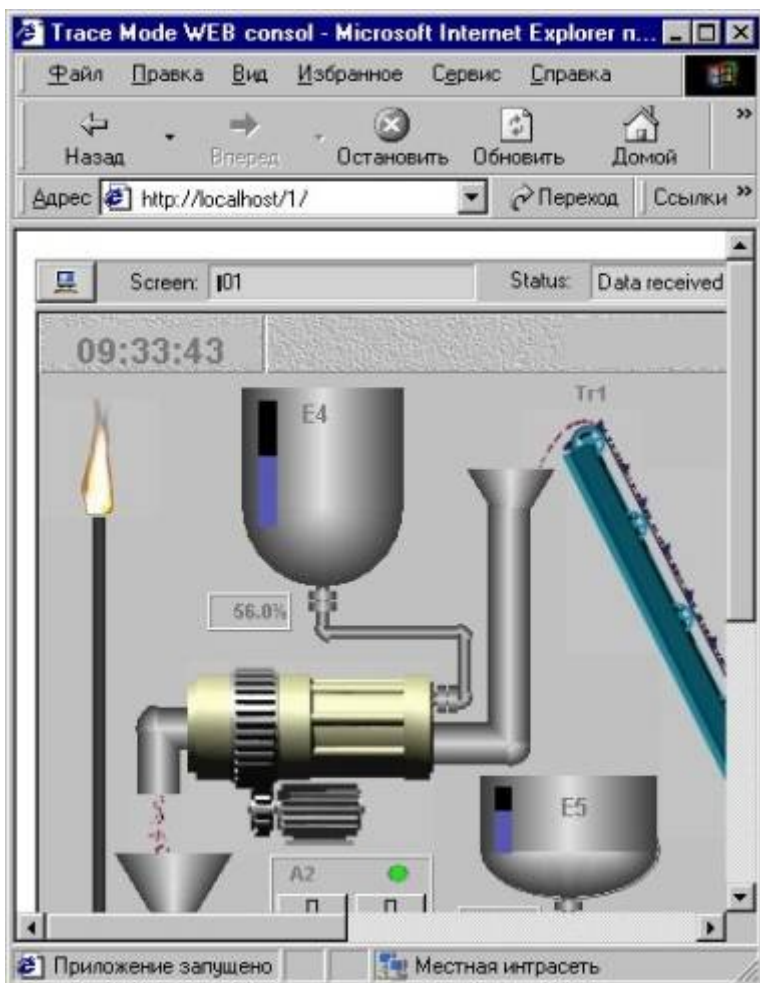
**anim\_period** – период смены кадров видео и мигания в миллисекундах, значение по умолчанию – 233;

**def\_font\_name** – имя шрифта (используется пока только в окне ОТ), по умолчанию – "Dialog";

**def\_font\_size** – размер шрифта, по умолчанию – 12.


## Доступ к проекту через Интернет

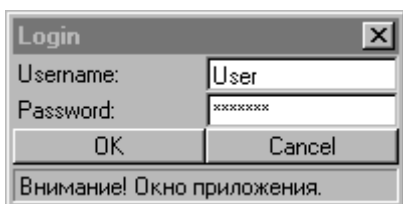
При обращении к созданной Web-странице по заданному для нее адресу статические ресурсы передаются удаленному пользователю. При этом в Web-браузере удаленного компьютера отображается графическая база узла:



**Внимание!** Для реализации данного режима рекомендуется использовать JVM либо от MS (IE 5,6), либо от Sun (Java 2 версия 1.4 или старше).

Адрес, указываемый в адресной строке web-браузера, должен заканчиваться знаком /. На рисунке показано задание адреса для случая, когда проект загружен в МРВ на локальной машине, а в качестве URL в диалоге **Настройка статических ресурсов** задано /1.

Чтобы подключиться к проекту, нужно нажать ЛК на иконке , ввести в появившемся диалоговом окне имя пользователя и пароль и нажать кнопку **ОК**:



Имена и пароли пользователей, а также их права на управление технологическим процессом задаются в редакторе базы каналов. Для этого нужно загрузить проект в

редактор и выполнить команду **Пароли** меню **Проект**. На экране появится диалог **Пользователи и права доступа**. Описание настроек этого диалога приведено в разделе [Паролирование](#) (см. **Работа в реальном времени**). В частности, для задания пользователю приоритета в управлении процессом через Интернет в этом диалоге следует установить флаг **GSM**.

Для окончания сеанса следует нажать ЛК на иконке  и закрыть окно Web-браузера.

**ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 13. [Разработка распределённых систем Trace Mode 5](#)**. Справочная система Trace Mode 5, раздел: «Разработка распределённых систем», главы: «Обмен по коммутируемым линиям», «Обмен по GSM», «Управление через интернет».

## ЛЕКЦИЯ 14

### БАЗЫ ДАННЫХ В ИСУ

1. Критерии оценки БД
2. Клиент-серверные технологии
3. Базы данных в промышленной автоматизации
4. Базы данных реального времени
5. Документирование в Trace Mode 5

- **организация документирования;**

- редактор шаблонов;

- **сервер документирования.**

В самом общем смысле база данных (БД) - это система хранения информации, обращение к которой осуществляется через средство управления базой данных (СУБД). На практике - это данные, рассортированные по уникальным идентификаторам и организованные в виде таблиц. Основное назначение БД - предоставить пользователю нужную информацию в нужном месте и в нужное время. И надо сказать, что по мере своего развития БД справлялись с этой задачей все лучше и лучше. Тем не менее, первые БД не совсем соответствовали ожиданиям. Организации и предприятия должны были бороться с огромными объемами дублированной и иногда противоречивой информации, предоставляемой, к тому же, различными и, зачастую, несовместимыми друг с другом способами.

#### **От прошлого к настоящему**

Можно сказать, что путь развития БД - это путь все большего и большего отстранения программного обеспечения от физических структур данных. До появления БД информация хранилась в отдельных файлах. Самые первые системы управления файлами позволяли программистам создавать, записывать, обновлять и читать эти файлы. Файловая система имеет органический недостаток: программы должны точно "знать", где расположены данные. Как следствие - для определения адресов в развитых системах хранения данных необходимо применение довольно сложных, трудно оптимизируемых и модифицируемых алгоритмов.

Первыми попытками абстрагирования программ от физических структур данных были индексные файлы, обеспечивающие доступ к информации посредством индексных ключей, т. е. для поиска записей в файле использовалась совокупность указателей. Такой подход решал определенный круг проблем, но индексным файлам по-прежнему были присущи многие ограничения, характерные для простых структур с единственной точкой входа. Сюда можно отнести, в частности, и неоптимальное хранение информации (дублирование, недостаточное структурирование), и значительное время поиска в больших файлах.

В качестве возможного решения этих проблем явились иерархические БД. В таких базах элементы данных строго упорядочены, причем так, что данные одного уровня подчиняются (является подмножеством) данным другого, более высокого уровня. В такой модели связи данных могут быть отражены в виде дерева-графа, где допускаются только односторонние связи от старших вершин к младшим. Иерархические БД не получили

широкого распространения. Реальный мир отнюдь не является иерархическим. Перспективнее оказались сетевые СУБД, учитывающие более сложные взаимосвязи между элементами, составляющими БД (теоретически, по крайней мере, допускаются связи "всех со всеми"). Управляющие программы для таких СУБД становились все более и более независимыми от физических структур данных. Но все равно необходимо знать, как управлять этими структурами. По-прежнему для таких моделей характерна сложность реализации СУБД, а сами программы остаются весьма чувствительными к модификациям. А поскольку каждый элемент данных должен содержать ссылки на другие элементы, требуются значительные объемы памяти, как дисковой, так и оперативной. Дефицит последней может приводить к замедлению доступа к данным, лишая сетевую БД основного ее достоинства - быстродействия.

Процесс отделения программ от структур данных в конечном итоге завершили реляционные базы данных (РБД). В РБД все данные представлены исключительно в формате таблиц или, по терминологии реляционной алгебры, отношений (relation). Таблица в реляционной алгебре - это неупорядоченное множество записей (строк), состоящих из одинакового набора полей (столбцов). Каждая строка характеризует некий объект, каждый столбец - одну из его характеристик. Совокупность таких связанных таблиц и составляет БД, при этом таблицы полностью равноправны - между ними не существует никакой иерархии. Реляционная модель является простейшей и наиболее привычной формой представления данных. РБД позволили моделям данных отражать взаимосвязи прикладной области, а не методы программного доступа к данным и структурам данных. Это - огромный шаг вперед по нескольким причинам:

- Отражающие прикладную область знаний модели данных являются интуитивно понятными конечному пользователю.
- Реорганизация данных на физическом уровне совершенно не влияет на выполнение прикладных программ. Одним из важнейших побочных эффектов данного преимущества является появление клиент-серверных архитектур, сохраняющих все достоинства централизованного администрирования и управления данными, с одной стороны, и дружески настроенных по отношению к пользователю клиентских программ, с другой.
- Благодаря нормализации удается избежать чрезмерного дублирования данных.

Индустрия РБД в настоящее время вполне созрела. Условия на рынке сейчас диктует "большая пятерка": IBM, Informix, Microsoft, Oracle и Sybase. На нее падает львиная доля всех расходов на разработку БД. Можно выделить две категории приложений в БД: оперативная обработка транзакций (OLTP - Online Transaction Processing) и системы поддержки принятия решений (DSS - Decision Support System).

OLTP-системы используются для создания приложений, поддерживающих ежедневную активность организации. Обычно это критические для деятельности приложения, требующие быстроты отклика и жесткого контроля над безопасностью и целостностью данных.

DSS (Decision Support System)-системы поддержки принятия решений, как правило, крупнее, чем OLTP-системы. Обычно они используются с целью анализа данных и выдачи отчетов и рекомендаций. Пользователи должны иметь возможность конструировать запросы различной степени сложности, осуществлять поиск зависимостей, выводить данные на графики и использовать информацию в других приложениях типа электронных таблиц, текстовых процессорах и статистических пакетов. Еще более широкую поддержку в процессе принятия решений обеспечивают системы оперативной аналитической обработки (OLAP - Online Analytical Processing).

## 1. Критерии оценки БД



Базы данных будут продолжать развиваться, а объемы информации в компьютерах - расти. Усложнение производственных процессов, "интеллектуализация" контрольно-измерительных приборов, требования конечного пользователя относительно повышения объемов и качества информации делают это предположение особенно справедливым для промышленных условий.

Однако наиболее важные критерии оценки БД останутся теми же самыми, а именно:

- Повышает ли БД возможности конечных пользователей путем предоставления доступа к нужной информации в нужном месте и в нужное время?
- Обеспечивает ли БД требуемый уровень открытости и гибкости запросов?
- Легко ли сопровождать и использовать БД? Надежна ли она?
- Широко ли распространена БД и хорошо ли поддерживается ее технология большим числом независимых производителей программного обеспечения?
- Легко ли интегрировать БД с широким спектром иного программного обеспечения?
- Широк ли спектр возможных применений БД?
- Доступны ли по цене большинству пользователей аппаратные платформы, поддерживаемые БД?
- Приемлема ли сама БД по цене для большинства пользователей?

## **2. Клиент-серверные технологии**

Модель "клиент-сервер" в настоящее время стала доминирующей компьютерной архитектурой после того, как предприятия осознали преимущество объединения удобных персональных компьютеров с централизованными, надежными и отказоустойчивыми мэйнфреймами. Клиент-серверные системы одновременно используют вычислительную мощь как клиента, так и сервера, возлагая интенсивную обработку данных на сервер и оптимизируя сетевой трафик так, чтобы повысить общую эффективность работы (рис.6.1).

Для интерфейса в клиент-серверных системах используется SQL - язык структурированных запросов (Structured Query Language). Он представляет собой средство организации, управления и поиска информации в РБД. Широкое признание SQL приобрел благодаря таким своим характеристикам, как:

- независимость от поставщика;
- переносимость на разные компьютерные платформы;
- опора на реляционные принципы хранения информации;
- высокоуровневая англоязычная структура;
- интерактивное выполнение запросов;
- полнофункциональный язык БД;
- поддержка со стороны IBM, Oracle, Sybase, Microsoft и др.

Язык SQL поддерживается всеми крупными поставщиками серверов БД и огромным большинством производителей различных прикладных средств разработки и языков.

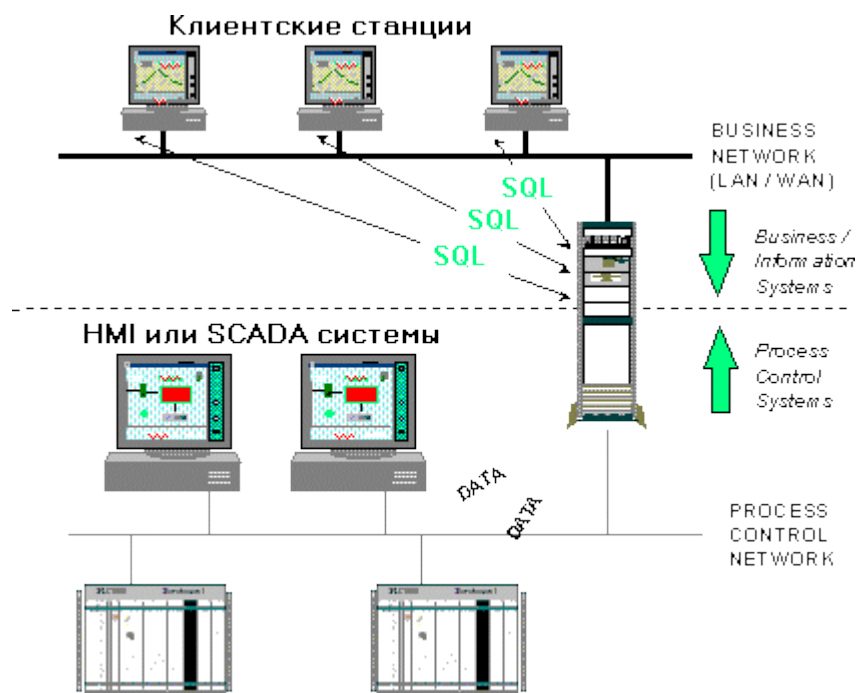


Рис. 6.1. Клиент-серверная организация.

### 3. Базы данных в промышленной автоматизации

С точки зрения организации информации заводская автоматизация несколько отстает от автоматизации офисной деятельности, при этом многие технологические и производственные БД основываются на устаревших и довольно негибких технологиях.

Как правило, производственному персоналу всегда не хватает информации. Операторам, специалистам, ремонтному персоналу, начальникам - всем нужен доступ к текущим и архивным производственным данным, статистической и итоговой информации и т.д. Все они хотели бы иметь какое-то единое средство доступа к информации, например, с мощью и открытостью РБД.

Однако, традиционные БД не всегда применимы в системах промышленной автоматизации. Можно выделить несколько основных ограничений:

- Производственные процессы генерируют данные очень быстро. Чтобы хранить производственный архив системы, например, с 7500 рабочими переменными, в БД каждую секунду необходимо вставлять 7500 строк. Обычные БД не могут выдержать подобную нагрузку.
- Производственная информация не вмещается! Многомесячный архив завода с 7500 рабочими переменными требует под БД дисковой памяти объемом около 1 Терабайта. Сегодняшние технологии такими объемами манипулировать не могут.
- SQL как язык не подходит для обработки временных или периодических данных, типичных для производственных систем. В частности, чрезвычайно трудно указать в запросе периодичность выборки возвращаемых данных.

#### 3.1. IndustrialSQL Server компании Wonderware

IndustrialSQL Server компании Wonderware позволяет преодолеть перечисленные ограничения, впервые превращая реляционную технологию в разумное решение для систем промышленной автоматизации. Что же такое IndustrialSQL Server?

IndustrialSQL Server - внутризаводской хранитель архивной информации, включая данные о событиях и соответствующих реакциях. IndustrialSQL Server представляет собой РБД, в которой учтена скорость поступления и объемы производственной информации.

Он позволяет осуществлять сбор и запись данных в сотни раз быстрее, чем это делают обычные БД на аналогичной платформе, и при этом еще и занимает значительно меньше дискового пространства.

IndustrialSQL Server - опора пакета промышленной автоматизации Wonderware FactorySuite2000. Несмотря на то, что IndustrialSQL Server поставляется компанией Wonderware как самостоятельный продукт, он, в то же время, является одним из главных компонентов пакета FactorySuite2000, являясь, можно сказать, его "сердцем". Будучи интегрированным со SCADA-компонентом InTouch, IndustrialSQL Server способен накапливать при помощи серверов ввода/вывода информацию практически от любых измерительных приборов и устройств сбора данных.

### **3.1.1. Взаимодействие – причина успеха**

IndustrialSQL Server - система управления РБД реального времени, использующая язык SQL. Выступая в качестве сервера БД, IndustrialSQL Server представляет собой расширение Microsoft SQL Server. При этом он обеспечивает скорость накопления данных более чем на порядок выше, характеризуется снижением размеров пространства хранения и реализует расширение языка SQL в области обработки данных, имеющих временные ярлыки (метки).

Объединение серверов IndustrialSQL Server и Microsoft SQL Server незаметно для пользователя. Можно сказать, что IndustrialSQL Server превращает Microsoft SQL Server в сервер РБД реального времени. При этом клиенты могут напрямую обращаться к IndustrialSQL Server при помощи тех же утилит, что и используются сервером Microsoft SQL Server.

Выбор Microsoft SQL Server в качестве основы для IndustrialSQL Server объясняется несколькими причинами. Во-первых, в мире существует более 200 миллионов пользователей Microsoft SQL Server. Во-вторых, Microsoft SQL Server является самой продаваемой БД для Windows NT. В-третьих, SQL поддерживается всеми крупными производителями серверов БД и большинством средств разработки и языков программирования.

Что делает IndustrialSQL Server с точки зрения взаимодействия IndustrialSQL - MS SQL?

- Сохраняет не критичную во времени информацию в БД Microsoft SQL Server. Вся технологическая информация сохраняется в специальных таблицах расширения.
- Поддерживает пропускную способность, то есть обеспечивает сохранение огромных потоков информации с высокой разрешающей способностью.
- Поддерживает целостность данных, то есть обеспечивает запись больших объемов информации без потерь.
- Добавляет в Microsoft SQL Server свойства сервера реального времени.

На рис. 6.1.1 показаны информационные потоки в системе управления. С одной стороны это данные, поступающие из различных источников для сохранения в БД, с другой - данные, запрашиваемые потребителями через интерфейс SQL сервера.

Стандартным механизмом поиска информации на сервере IndustrialSQL Server является SQL, что гарантирует доступность данных самому широкому кругу приложений. В подмножество языка SQL входит расширение, служащее для получения динамических производственных данных из IndustrialSQL Server и позволяющее строить запросы на базе временных отметок. Все приложения, работающие с Microsoft SQL Server, могут также подключаться и к IndustrialSQL Server.



Рис. 6.1.1. IndustrialSQL Server на основе MS SQL Server.

Используемая в IndustrialSQL Server архитектура клиент-сервер позволяет заполнить промежуток между промышленными системами контроля и управления реального времени, характеризующимися большими объемами информации, и открытыми гибкими управленческими информационными системами. Благодаря наличию мощного и гибкого процессора запросов пользователи имеют возможность осуществлять поиск любой степени сложности для выявления зависимостей и связей между физическими характеристиками, оперативными условиями и технологическими событиями.

### 3.1.2. Характеристика РБД IndustrialSQL Server. Функциональные возможности

Высокопроизводительный сервер.

IndustrialSQL Server обеспечивает сбор данных в сотни раз быстрее, чем любые другие РБД, и сохраняет их на гораздо меньшем дисковом пространстве. Многоуровневая клиент-серверная архитектура служит мостом между управленческими и производственными сетями, предоставляя вышележащему уровню всю информацию в реальном масштабе времени. Опирающаяся на Windows NT Server многоуровневая архитектура представляет собой масштабируемое решение любых пользовательских требований. IndustrialSQL Server может использоваться как в небольших цехах с сотней регистрируемых технологических параметров, так и на крупных промышленных предприятиях с сотнями тысяч параметров.

Уменьшение объема хранения.

IndustrialSQL Server позволяет хранить данные на пространстве, составляющем небольшую долю от соответствующего объема обычной РБД. Фактический размер требуемого для хранения производственной информации дискового пространства определяется размером и сущностью операций предприятия, а также интервалом хранения предыстории его функционирования. Например, двухмесячный архив предприятия с 4000 параметров, опрашиваемых с периодичностью от нескольких секунд до нескольких минут, будет занимать около 2 Мб дискового пространства. Используемый алгоритм упаковки информации является алгоритмом сжатия без потерь, сохраняющим высокое разрешение и качество данных.

Достоверность информации.

Будучи сервером БД в составе пакета FactorySuite 2000, IndustrialSQL Server хранит наиболее полную информацию о производственных процессах. Сервер может накапливать производственную информацию с высокой разрешающей способностью, получая ее при помощи серверов ввода/вывода от более чем 600 различных контрольных и регистрирующих устройств, а также от станций InTouch и системы ввода/вывода InControl. Все эти данные объединяются сервером с конфигурационной, аварийной, итоговой информацией, сведениями о событиях, архивом InBatch, информацией системы контроля перемещения InTrack и прочими технологическими данными.

Объединение данных предоставляет пользователю множество преимуществ, выводя его на новый уровень представления о состоянии и ходе производственного процесса. Такой объем информации может быть полезен лишь тогда, когда пользователь имеет на руках мощный процессор запросов, позволяющий обрабатывать и фильтровать необходимые данные. IndustrialSQL Server обладает всей мощью Microsoft SQL Server со всеми его средствами фильтрации, объединения и обработки данных.

Конфигурационные параметры, как и вся предыстория модификаций, хранятся в "чисто" Microsoft SQL - таблицах, доступных через SQL. В процессе функционирования предприятия могут добавляться новые и удаляться существующие параметры, меняться описания и диапазоны измерений. Сохранение предыстории модификаций гарантирует соответствие конфигурационных параметров возвращаемым сервером архивным данным.

### **Сервер реального времени**

В язык запросов IndustrialSQL Server включены средства работы с временными характеристиками данных. Входящие в состав Wonderware FactorySuite серверы ввода/вывода используют новый протокол SuiteLink. В этом протоколе впервые была введена концепция отметок времени и качества информации, выставляемых серверами ввода/вывода. Кроме того, благодаря протоколу SuiteLink удалось еще более повысить скорость накопления информации.

### **Система регистрации событий**

Непрерывные данные наиболее полезны в контексте событий. Событие может представлять собой все, что угодно - завершение серии, изменение значения переменной, операции SQL по вставке, обновлению или удалению, заступление новой смены либо запуск оборудования и т.д., а также комбинации всего перечисленного. IndustrialSQL Server может различать и соответствующим образом реагировать на события. События могут инициировать определенные предписанные действия. Например, завершение очередного этапа может приводить к записи конечных значений этапа в таблицу серии, начало новой смены может запустить выдачу сменного отчета, запуск двигателя может привести к посылке определенного сообщения в ремонтную службу и т.д. Функции копирования облегчают тиражирование сводных данных и информации о событиях, что особенно важно при принятии различных управленческих решений.

### **Гибкий открытый доступ**

Большая доля производственной информации имеет такие же характеристики, как и обычные деловые данные (например, конфигурационные или сводные данные). Информация подобного рода поддерживается средствами Microsoft, встроенными в IndustrialSQL Server, а именно, сервером Microsoft SQL Server. В производственных отчетах, как правило, содержится сводная (статистическая) информация. IndustrialSQL Server может автоматически обновлять сводные таблицы с заданной периодичностью,

записывая в них средние величины, суммы, а также максимальные и минимальные значения.

А имеющиеся клиентские приложения дают пользователям возможность выбирать именно те средства, которые наилучшим образом позволяют решать поставленные задачи. Хотя методы доступа и являются стандартными, безопасность данных никоим образом не ущемляется. IndustrialSQL Server опирается на средства ограничения несанкционированного доступа систем Microsoft SQL Server и Windows NT, гарантируя тем самым требуемый уровень защиты информации. IndustrialSQL Server представляет собой единственное место доступа к производственной информации и единую платформу разработки прикладных приложений для производства и связи с управленческими системами. Регистрация в системе, поддержание групп пользователей и управление доступом к БД упрощается благодаря Microsoft SQL Enterprise Manager.

### **SQL с поддержкой временных параметров**

Обычный язык SQL не поддерживает временные характеристики данных. В частности, в нем нет никаких средств контроля времени поступления данных и никакого способа предоставления клиенту не запрошенных данных. IndustrialSQL Server расширяет возможности Transact-SQL, являющегося реализацией SQL для Microsoft SQL Server, обеспечивая управление разрешением и обновлениями, а также предоставляя основу таким временным функциям, как частота изменения и интегральные вычисления на сервере.

### **Простота конфигурирования**

Одними из достоинств IndustrialSQL Server являются наличие готового набора функциональных возможностей и быстрота его установки в рабочей системе. Все выполняется простым нажатием на кнопку мыши, при этом сервер определяет собственные параметры с учетом существующего InTouch-приложения.

### **Открытая и гибкая база данных**

Мощная и гибкая БД IndustrialSQL Server поддерживает доступ к информации реального времени, архивным и конфигурационным данным любыми программными средствами. Для хранения информации доступны следующие типы данных (рис.6.1.2):

- реального времени;
- архивные;
- конфигурационные;
- сводные;
- сопутствующие учрежденческие.

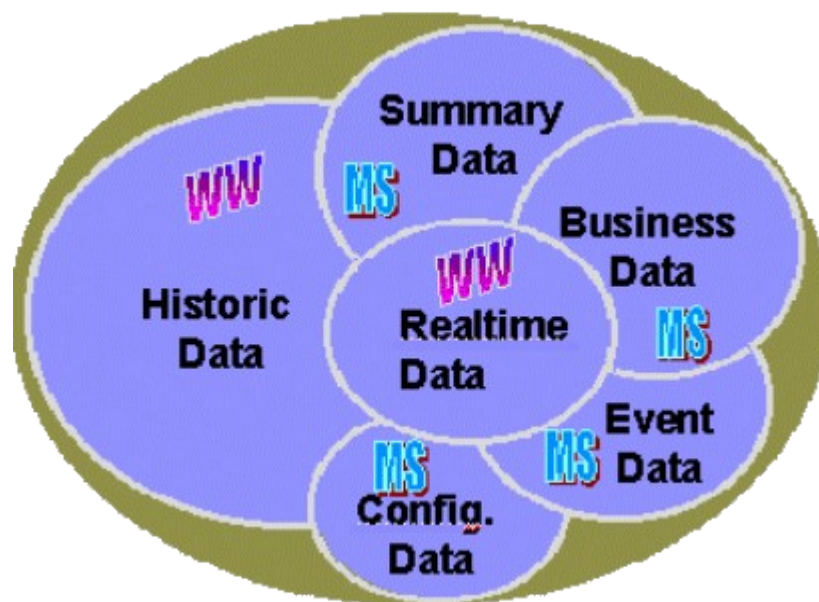


Рис. 6.1.2 Типы данных, регистрируемых IndustrialSQL Server

Идеология построения таблиц РБД, интегрирующих столь разнообразные типы данных из различных источников, имела ориентацию на улучшение характеристик производительности, качества и стоимости в таких ключевых областях как:

- анализ протекания процесса, диагностика, оптимизация;
- управление запасами: потребление сырья;
- техническое обслуживание (предупредительные и превентивные ремонты);
- продукция и контроль качества (SPC/SQC);
- функционирование в качестве системы управления производственным процессом

### **Простота использования**

Для установки, конфигурирования и использования IndustrialSQL Server от пользователя не требуется никакого знания языка SQL. Особенностью IndustrialSQL Server является его ориентация на готовые наборы функций. IndustrialSQL Server разрабатывался как не требующая никакого администрирования система управления БД. Резервные копирования базы могут выполняться средствами Microsoft BackOffice. Наличие сотен клиентских приложений позволяет выбирать из них именно то, которое соответствует требованиям пользователя по простоте и функциональным возможностям.

### **Интеграция с другими компонентами пакета FactorySuite 2000**

Будучи БД в составе FactorySuite 2000, IndustrialSQL Server тесно связан с любым компонентом этого пакета на любом уровне. Конфигурационные данные SCADA-системы InTouch хранятся вместе с конфигурационными данными IndustrialSQL Server. IndustrialSQL Server получает данные от серверов ввода/вывода, DDE, FastDDE и SuiteLink, а также хранит архивы InTouch, InControl, InBatch, InTrack и SPCPro. Для просмотра данных и построения аналитических графиков InTouch может использовать как собственные архивы, так и архивы IndustrialSQL Server. Кроме того, для построения графиков в нем может использоваться новый ActiveX-объект ActiveTrend, а для извлечения данных IndustrialSQL Server, так и ActiveX-объекты доступа к базам данных, разработанные сторонними производителями. Браузер Scout имеет возможность читать данные IndustrialSQL Server. Для работы с IndustrialSQL Server были разработаны средства FactoryOffice и IndustrialWorkbook. А компоненты InControl и InTouch имеют возможность выступать в качестве поставщиков информации для IndustrialSQL Server.

### 3.1.3. Области применения

В перечень обязанностей производственно-технического персонала предприятия входят повышение качества продукции, повышение эффективности производства, а также повышение коэффициента полезного действия используемого оборудования. Все эти цели недостижимы без владения оперативной и архивной информацией о состоянии производства и характеристиках выпускаемой продукции.

Специалисты по контрольно-измерительным средствам должны иметь полную информацию о структуре и функционировании всей системы контрольно-измерительных приборов. IndustrialSQL Server может предоставить им всю необходимую конфигурационную информацию типа значений контрольных параметров, допустимых ошибок и предельных границ, а также осуществлять регистрацию функционирования всей системы, записывая информацию типа отклонений рабочих параметров от установленных, ошибок измерения и выходов за предельные границы и, тем самым, позволяя находить ответы на вопросы типа: Является ли значение данной контрольной точки оптимальным для данного контура регулирования? Не привело ли срабатывание блокировочного узла к генерации ложной ошибки? Достаточен ли объем информации, выдаваемой оператору данным алармом?..

Технологический персонал должен иметь информацию о поведении процесса в установившемся и неустойчивом режиме. IndustrialSQL Server хранит всю информацию о параметрах и событиях процесса, предоставляя специалистам возможность анализировать переходные и аварийные состояния процесса.

Обслуживающий персонал должен иметь информацию о текущем состоянии оборудования и условиях его эксплуатации. IndustrialSQL Server хранит как производственный архив, так оперативные данные.

Руководителя производственных отделов нуждаются в итоговой информации о ходе производственного процесса и основных событиях. IndustrialSQL Server может предоставлять требуемые данные, как в итоговом, так и сгруппированном виде, а также записывать информацию о произошедших событиях. С его помощью руководители смогут получать точные ответы на такие вопросы типа: Каков объем дневного выпуска продукции? Каковы причины и длительность простоев оборудования в этом месяце? Соответствует ли выпуск продукции плановым показателям?..

Работники службы контроля качества должны иметь полную информацию о качестве выпускаемой продукции, несоответствиях и отклонениях от заданных параметров. IndustrialSQL Server может осуществлять запись всех измеряемых технологических параметров и связывать их с конкретной продукцией либо партией, помогая находить ответы на вопросы типа: Не повлияло ли изменение технологической карты на качество продукции? Какова вероятность появления дефектов в продукции данного типа? Существует ли взаимосвязь между данным температурным профилем и отклонениями данного параметра от заданного значения?..

Операторы технологического оборудования должны иметь возможность сравнивать текущие условия эксплуатации с существовавшими ранее и выявлять аномальное поведение процесса. IndustrialSQL Server хранит как оперативные, так и архивные данные и позволяет сравнивать их.

## 3.2. Plant2SQL и новые возможности, предлагаемые компанией Ci Technologies

Родственный Citect продукт, называемый Plant2SQL, позволяет предоставлять технологическую информацию, являющуюся прерогативой SCADA-систем.

Plant2SQL поддерживает простой доступ к данным технологического процесса как из приложений, так и со стороны пользователей. Пользователям теперь доступна самая



последние данные технологического процесса, что позволит им принимать решения во всеоружии, полностью владея информацией о процессе производства.

Большинство SCADA-систем имеет возможность обмениваться данными с множеством баз данных, однако, если необходимо выполнить какие-то модификации в алгоритме обмена данными, то возникают проблемы. Обычно персонал уровня управления предприятием не хочет знать особенности SCADA-систем. С появлением Plant2SQL нет необходимости управляющему персоналу предприятия знать SQL или особенности получения данных из SCADA-архивов.

### 3.2.1. Основные особенности Plant2SQL

Открытые технологии, такие, как Microsoft ActiveX, используются для упрощения интеграции Plant2SQL с пакетами, такими, как Microsoft Word, Excel, Access, Internet Explorer, Visual Basic.

Основные особенности Plant2SQL:

- легкий доступ к технологическим данным;
- открытые базы данных;
- никакой конфигурации или модификации в Citect не требуется;
- поддержка резервирования;
- не требуется знания SQL языка;
- установка и просмотр данных выполняется несколькими нажатиями кнопки мыши;
- простой выбор выбранных пользователем данных для просмотра;
- адаптируемость и расширяемость;
- клиенты могут читать данные из баз данных SQL или прямо из SCADA-системы.

На основе стандартных протоколов осуществляется обмен данными в Plant2SQL (см. рис. 6.2.1)

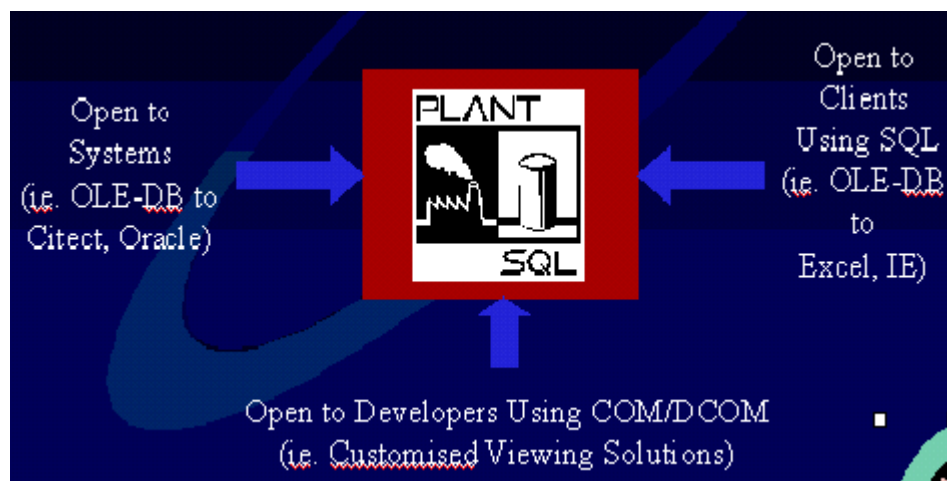


Рис. 6.2.1 Протоколы доступа к Plant2SQL

### Клиентские приложения Plant2SQL

Plant2SQL включает ряд клиентских приложений, которые могут настраиваться на различные требования пользователей.

Одно из таких приложений поставляется для Microsoft Excel. Оно позволяет пользователю выбирать данные и встраивать их в электронные таблицы. При встраивании допустимо использование всех стандартных средств (tools), чтобы представлять и анализировать информацию, а затем сохранять ее для повторного использования.

Сбор данных. Plant2SQL представляет простые и быстрые средства конфигурирования для обеспечения сбора данных.

Plant2SQL легко интегрирует данные технологического процесса в существующий или новый SQL Server. Если SQL Server не устанавливается, то Plant2SQL будет сохранять информацию, используя Microsoft Data Engine (MSDE), который поставляется с Plant2SQL и является на 100% совместимым с Plant2SQL (рис. 6.2.2).

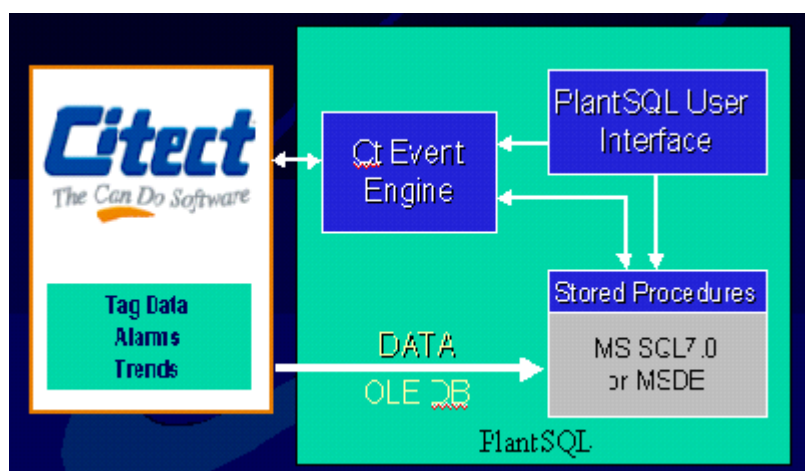


Рис.6.2.2 MS SQL Server - основа Plant2SQL

По умолчанию все трендовые и алармовые данные автоматически доступны клиентскому приложению. Пользователи могут только отметить точки, которые необходимо зарегистрировать в базе данных Microsoft SQL и иметь доступ к отдельным точкам.

Plant2SQL включает подсистему событий, которая просматривает события в Citect и может быть использована, чтобы запускать передачу или хранение набора данных. В Plant2SQL этот набор данных называется Snapshot (снимок). Мгновенные выборки переменных (Snapshots) активизируются из множества источников, включая определенные моменты времени или условные выражения переменных в Citect. Каждая выборка может быть сконфигурирована, чтобы включать любую группу переменных с возможной записью в эти переменные.

**Архитектура.** Plant2SQL имеет различные опции расширения. В малых простых приложениях возможен запуск Plant2SQL сервера и клиента на одном компьютере как клиент и сервер Citect. Если приложение растет, то разные компьютеры могут использоваться для Citect, для Plant2SQL сервера, Plant2SQL клиента и даже отдельный файл-сервер для базы данных, если потребуется.

**Резервирование.** Plant2SQL имеет встроенные средства резервирования. Отдельный Plant2SQL может подключаться к основному Citect-серверу и автоматически переключаться на резервный Citect-сервер при возникновении проблем с основным.

Если необходима резервная база данных SQL Server, то стандартные средства репликации могут быть использованы для репликации базы данных в резервный SQL Server.

Если необходимы резервные Plant2SQL серверы, то пара Plant2SQL серверов может быть подключена к паре Citect серверов.

**Замечание.** В Plant2SQL не существует синхронизации между основной и резервной базами данных Plant2SQL.

Plant2SQL клиенты позволяют не техническим пользователям получать данные. В некоторых случаях может потребоваться более высокая степень гибкости, и Plant2SQL обеспечивает это как серверу, так и клиенту.

На стороне сервера Plant2SQL обеспечивается хранимыми процедурами (stored procedures), которые автоматически устанавливаются в SQL Server или MSDE. Plant2SQL использует эти хранимые процедуры, чтобы получать данные из Citect и сохранять их в SQL сервере или MSDE. Эти же хранимые процедуры доступны через

документированный интерфейс. Например, возможно писать собственные хранимые процедуры и вызывать хранимые процедуры Plant2SQL для доступа к данным из Citect.

С клиентской стороны Plant2SQL обеспечивается ActiveX интерфейсом, который доступен любому приложению.

Plant2SQL с MSDE или SQL сервером. Plant2SQL предлагает выбор между Microsoft MSDE и SQL Server 7.0. MSDE является частью SQL Server. Для многих приложений MSDE будет вполне достаточен. MSDE имеет меньший footprint (85 MB), но ограничивается 2 GB на базу данных и оптимизирован, когда количество одновременно работающих клиентов не превышает 5. Производительность сильно падает при увеличении количества пользователей. Основное ограничение - 2 GB на область хранения.. Но так как Plant2SQL поддерживает гетерогенные запросы, то количество требуемого пространства минимизируется.

### **3.2.2. Область применения**

Интеграция заводских данных с бизнес-информацией открывает большие возможности для улучшения деятельности предприятия, качества и производительности.

Персонал отдела качества (Quality Assurance) может легко сравнить продукцию производства со спецификацией, проанализировать качество.

Отдел поддержки (обслуживающий персонал) количество часов работы оборудования, планируемую диагностику оборудования.

Менеджеры по производству могут легко интегрировать бизнес-информацию с технологической и быстро просчитывать стоимость инвестиций и материальных издержек.

## **4. Базы данных реального времени. Чем же они отличаются!**

Рассматриваемые БДРВ в качестве основы используют одну из распространенных БД Microsoft SQL Server (следует напомнить, что имеют место и другие решения). Преимущества такого подхода следующие:

- большое количество пользователей владеют продуктом и потому в проектных решениях могут использовать не только возможности БДРВ, но и создавать собственные базы данных или таблицы в рамках существующей базы данных реального времени;
- новые технологические решения (например, OLE DB), предлагаемые Microsoft и реализуемые в MS SQL Server не требуют серьезных вложений со стороны поставщиков БДРВ. Проведение адаптации возможностей MS SQL Server для БДРВ сокращает сроки появления новых версий БДРВ с новыми возможностями;
- техническое сопровождение упрощается.

Как видно на примере указанных БД, несмотря на то, что в основе лежит MS SQL Server, реализованы они различно:

- для хранения данных реального времени в IndustrialSQL Server используются исторические блоки или файлы специального формата. Основное требование к ним - обеспечение высокой скорости регистрации и повышенное сжатие данных. В Plant2SQL технологические данные хранятся в стандартных MS SQL таблицах. Для обеспечения высокой скорости регистрации используется стандартная подсистема архивов Citect;
- IndustrialSQL Server обеспечивает регистрацию в реальном времени из серверов ввода-вывода по протоколам DDE, OPC, SuiteLink. Режим регистрации в Plant2SQL поддерживается либо системой архивирования Citect, либо, используя API (Application Programming Interface) для произвольных приложений Windows;

- Доступ из клиентских приложений осуществляется по SQL-запросам. В IndustrialSQL Server в версии 7.1 добавлена возможность получения по DDE, SuiteLink-протоколам.

## 5. Документирование в Trace Mode 5

### Организация документирования

Документирование технологической информации – это одна из основных функций систем управления. Документы должны соответствовать требованиям к технологическим отчетам и журналам, принятым на производстве. Они могут существенно отличаться на разных предприятиях. Поэтому для решения задачи документирования необходимо иметь инструмент создания произвольных форм документов.

В ТРЕЙС МОУД для документирования технологической информации используется специальный модуль – [сервер документирования](#). Этот модуль по команде от мониторов реального времени, собственному сценарию или по команде от оператора интерпретирует созданные заранее шаблоны, запрашивает у МРВ необходимые данные и формирует по ним готовые документы. Для создания шаблонов документов в инструментальную систему включен специальный редактор – **редактор шаблонов**.

Шаблон документа разрабатывается в виде файла HTML-формата. В него могут быть вставлены любые элементы, поддерживаемые в HTML, а так же дополнительные функции и команды, предназначенные для запроса данных от узлов проекта ТРЕЙС МОУД и обработки полученных значений.

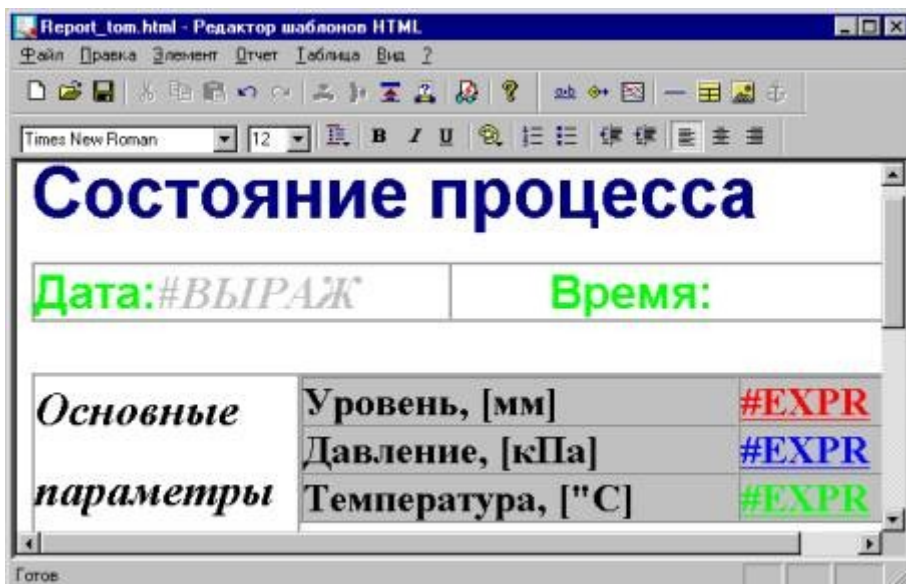
Команды, которые доступны для использования в шаблонах, позволяют выводить значения атрибутов каналов в нужных областях генерируемого документа, вставлять в него растровые изображения, данные из архивов ТРЕЙС МОУД в виде графиков, интегральных и усредненных значений каналов.

### Редактор шаблонов

- Для разработки шаблонов документов в состав инструментальной системы включен **Редактор шаблонов**. Чтобы запустить редактор, нужно дважды нажать ЛК на соответствующем ярлыке в программной группе **Трейс Моуд**. Другой способ – запустить файл **htmpled.exe**, который находится в директории инструментальной системы.

### Описание интерфейса редактора

Экран редактора шаблонов представлен на следующем рисунке.



Редактор имеет главное меню, панели инструментов, строку состояния и рабочее поле.  
Главное меню редактора шаблонов

[Файл](#)

[Правка](#)

[Элемент](#)

[Отчет](#)

[Таблица](#)

[Вид](#)

[Помощь](#)

Главное меню содержит следующие разделы:

Файл;

Правка;

Элемент;

Отчет;

Таблица;

Вид;

Помощь.

Нажатие ЛК на любом из них открывает соответствующее меню.

## Файл

В меню **Файл** включены команды работы с файлами шаблонов, печати документа, а также команда выхода из редактора. Ниже приводится описание этих команд.

**Создать** – создать новый шаблон. Новому шаблону присваивается имя **Безымянный**. Это имя можно затем изменить при сохранении шаблона.

**Открыть** – загрузить в редактор существующий файл шаблона. Эта команда выводит на экран стандартный диалог выбора файла.

**Сохранить** – охранить шаблон с тем же именем, под которым он был загружен.

**Сохранить как** – сохранить шаблон в файл под другим именем.

**Выбрать проект** – эта команда позволяет связать шаблон с проектом ТРЕЙС МОУД для вставки команд вывода данных реального времени и архивных данных.

**Параметры документа** – эта команда открывает диалоговое окно настройки следующих параметров документа: заголовок, интервал обновления при просмотре в браузере [сек], цвет фона, фоновый рисунок. Здесь можно указать, является ли документ фрагментом, и настроить параметры этого фрагмента.

**Печать** – вывод загруженного шаблона на печать.

**Выход** – завершить работу редактора шаблонов.

## Правка

В меню **Правка** включены команды редактирования текста шаблонов. Они выполняют следующие действия:

**Отменить** – отмена последних действий (Undo).

**Вернуть** – восстановление отмененных действий (Redo).

**Вырезать** – удалить выделенный текст и поместить его в буфер обмена.

**Копировать** – копировать выделенный текст в буфер обмена.

**Вставить** – вставить в текущую позицию курсора текст из буфера обмена.

**Удалить** – удалить выделенный текст.

**Выделить все** – выделить весь текст шаблона.

**Вставить фрагмент** – вставить в текущую позицию курсора фрагмент документа из файла.

## Элемент

Данное меню предназначено для работы с отдельными элементами текста шаблона и функциями, выполняемыми сервером документирования при подготовке документа. Оно включает в себя следующие команды и разделы.

**Вставить** – раздел, содержащий команды вставки следующих элементов:

Рисунок.

Метка перехода.

Гиперссылка.

Произвольное выражение.

Имя канала.

Значение канала.

Значение времени.

Статистическое выражение.

Окно тренда.

Подробное описание вставки этих элементов приведено ниже в соответствующих разделах данного описания.

**Линейная группировка** – объединение выделенных линейных элементов языка HTML в один элемент.

**Блочная группировка** – объединение выделенных блочных элементов языка HTML в один элемент.

**Обобщить выбор** – выделить элементы, находящиеся в иерархии на ступень выше.

**Свойства элемента** – вывод на экран диалога редактирования свойств текущего HTML-элемента.

**Метки перехода** – вывод списка меток в текущем документе и переход к выбранной метке.

**Пользовательские функции** – вывод на экран диалога создания и редактирования пользовательских функций.

Линейными называются элементы, не прерывающие текущий абзац. Это, например, элементы управления параметрами шрифта, вставки изображений, динамические выражения ТРЕЙС МОУД, ActiveX и гиперссылки.

Блочные элементы обязательно располагаются в отдельных абзацах. Такими элементами являются, например, абзац, таблица, список.

**Отчет**

Это меню содержит одну команду – **Пробный отчет**. При ее выполнении на экране появляется окно, в котором выводится изображение сгенерированного по шаблону документа. Этот документ может быть выведен на принтер как образец. Однако данные, которые вставлены в него, генерируются редактором шаблонов, а не запрашиваются у мониторов реального времени.

## **Таблица**

Это меню содержит команды для создания и редактирования таблиц, вставляемых в документ.

**Вставить таблицу** – вставить новую таблицу в текущую позицию курсора.

**Вставить строку** – добавить строку в таблицу. Созданная строка будет расположена над текущей.

**Вставить столбец** – добавить столбец в таблицу. Созданный столбец будет расположен слева от текущего.

**Вставить ячейку** – добавить ячейку в таблицу. Созданная ячейка будет расположена слева от текущей.

**Удалить строки** – удаление текущей строки таблицы.

**Удалить столбцы** – удаление текущего столбца таблицы.

**Удалить ячейки** – удаление текущей ячейки таблицы.

**Склеить ячейки** – объединить выделенные ячейки в одну.

**Разделить ячейку** – разбить текущую ячейку на две. Ячейка разбивается в пределах строки

## **Вид**

Это меню предназначено для управления внешним видом окна редактора шаблонов. Оно содержит следующие команды.

**Панели инструментов** – раздел управления панелями инструментов. Он содержит три команды управления видимостью соответствующих панелей:

**Основная;**

**Форматирования;**

**Объектов.**

Видимые панели отображаются в меню символом "(" левее названия команды.

**Строка состояния** – управление видимостью строки состояния.



**Скрытые элементы** – управление видимостью вспомогательных элементов при редактировании.

**Показать рамки** – управление видимостью при редактировании рамок таблиц, объявленных невидимыми.

## **Помощь**

Это меню позволяет получить информацию о редакторе шаблонов и войти в справочную систему.

## **Сервер документирования**

Для документирования технологической информации в ТРЕЙС МОУД предусмотрен специальный модуль – сервер документирования. Существует два типа таких серверов: локальный и глобальный. Первый из них может получать данные для документирования только от одного МРВ, к которому подключается локально (не по сети), и выдавать документ только на принтер. Глобальный сервер документирования может работать со всеми узлами проекта, объединенными в локальную сеть, сохранять документ в файл и публиковать его на WEB сервере.

Чтобы глобальный сервер документирования мог найти мониторы реального времени, к которым он подключается по сети, ему надо сообщить имена/IP-адреса машин, на которых МРВ запущены. Для задания имени/IP-адреса компьютера используется соответствующая вкладка диалога **Параметры узла**.

Информацию об ошибках связи с МРВ сервер документирования записывает в файл **dslog<MMDD>.txt**, где <MMDD> – месяц и день создания файла.

Далее описана работа и настройка глобального сервера документирования. Это описание полностью соответствует работе локального сервера за исключением указанных выше ограничений.

## **Организация работы сервера**

Создание документов осуществляется по шаблонам, которые подготавливаются в **редакторе шаблонов**. Время или условие генерирования, имя файла шаблона, а также направление вывода документа описываются в программах документирования – **сценариях**.

## **Направления документирования**

Создаваемые документы могут либо выводиться на принтер, либо сохраняться в файлы HTML-формата, а также могут публиковаться на WEB-сервере.

Форма выдачи документа называется направлением.

## **Генерация документов по времени**

Чаще всего подготовка отчетов привязывается к астрономическому времени. Так, например, они могут генерироваться раз в час, раз в сутки, раз в месяц и пр.

Для решения подобных задач в сценариях сервера документирования задается дата и время подготовки документа. Дата в сценарии может отсутствовать. В этом случае генерирование документа будет осуществляться каждый день в указанное время.

Существует возможность настроить генерирование документа с необходимым периодом, например, раз в десять минут. Кроме того, можно установить режим подготовки документа раз в смену и затем описать разбивку суток на смены.

### Генерация документов по командам МРВ

Любой монитор МРВ в проекте может являться инициатором генерирования документа. Для этого при настройке сценария надо указать узел, который будет управлять запуском сценария, управляющий канал на этом узле и контрольное значение.

Если величина канала равна контрольному значению, то запускается генерирование документа. После этого сервер документирования обнуляет значение управляющего канала.

### Подготовка документов по команде

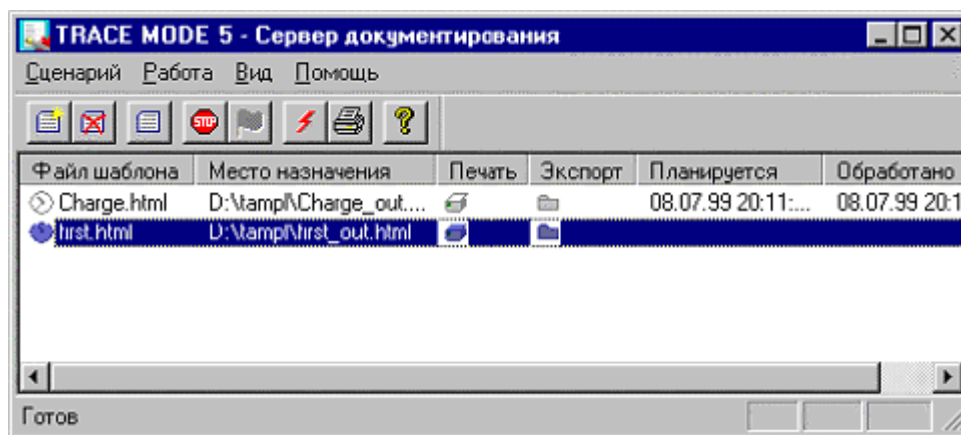
Кроме штатных ситуаций подготовки документов, которые можно описать в сценариях или предусмотреть в математике операторских станций, может возникнуть необходимость осуществить внеочередную генерацию документа. Для этого в сервере документирования предусмотрена возможность подготовки документа по команде оператора.

При генерировании документа по команде оператора надо выбрать сценарий, описывающий создание требуемого документа, и нажать специальную иконку на инструментальной панели сервера документирования.

### Запуск сервера документирования

Сервер документирования может быть запущен выбором соответствующего ярлыка в программной группе ТРЕЙС МОУД. Другой способ - запустить из командной строки `docserv.exe`.

При запуске сервера документирования на экран выводится его главный диалог, представленный на следующем рисунке.



Этот диалог содержит главное меню, инструментальную панель, список запрограммированных сценариев и строку состояния.

## Главное меню сервера документирования

[Сценарий](#)

[Работа](#)

[Вид](#)

[Помощь](#)

Главное меню содержит следующие разделы:

Сценарий;

Работа;

Вид;

Помощь.

Нажатие ЛК на любом из них открывает соответствующее меню.

### **Сценарий**

В меню **Сценарий** включены команды работы со сценариями, а также команда выхода из редактора:

**Создать** – создание нового сценария.

**Удалить** – удаление выделенного сценария.

**Настройка** – редактирование выделенного сценария.

**Выход** – завершение работы сервера документирования.

### **Работа**

В меню **Работа** включены команды управления выполнением сценариев. Они реализуют следующие действия:

**Приостановить** – запретить автоматическое выполнение текущего сценария.

**Возобновить** – возобновить автоматическое выполнение текущего сценария.

**Запустить сейчас** – сгенерировать документ по заданному в выделенном сценарии шаблону и направлениям.

**Напечатать сейчас** – вывести на принтер документ, сгенерированный по шаблону, заданному в выделенном сценарии.

### **Вид**

Это меню предназначено для управления внешним видом окна сервера документирования. Оно содержит следующие команды:

**Панель инструментов** – управление видимостью панели инструментов;

**Строка состояния** – управление видимостью строки состояния.

Видимые элементы отображаются в меню символом "(" левее названия команды.

## **Помощь**

Это меню позволяет получить информацию о текущей реализации сервера документирования и войти в справочную систему.

## **Сценарии**

Сценарии – это программы документирования. В них описывается время или условие генерирования документа, какой использовать для этого файл шаблона. В сценариях также прописываются направления вывода документов.

### **Создание сценария**

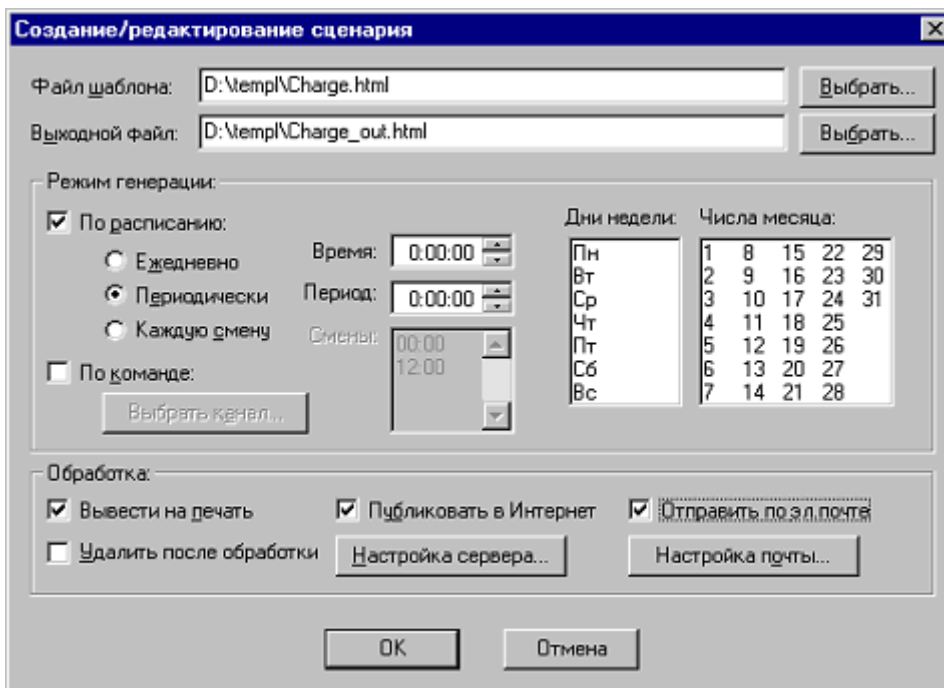


Чтобы создать сценарий, надо выполнить команду **Создать** из меню **Сценарий** или нажать ЛК на соответствующей иконке инструментальной панели. На экране появляется диалог, в верхней строке которого указывается файл шаблона документа и путь к нему. Вторая строка используется для выбора файла, в который будет сохраняться сгенерированный документ.


В разделе **Режим генерации** настраиваются условия выполнения сценария. Это может быть либо астрономическое время, либо команда от МРВ.

Раздел **Обработка** диалога предназначен для настройки направлений вывода. Здесь можно дать указание выводить документ на принтер, настроить его публикацию на WEB-сервере и отправку по электронной почте.

Вид диалога показан ниже на рисунке.



## Редактирование сценария

 Чтобы внести коррективы в уже существующий сценарий, следует сначала выделить его в списке. Затем надо выполнить команду **Настройка** из меню **Сценарий** или нажать ЛК на соответствующей иконке инструментальной панели.

После этого на экран будет выведен тот же диалог, что и при создании сценария. Однако в нем уже будут присутствовать параметры выбранного сценария.

## Удаление сценария

- Для удаления любого сценария надо сначала выбрать его в списке. Затем следует выполнить команду **Удалить** из меню **Сценарий** или нажать ЛК на соответствующей иконке инструментальной панели. После этого указанный сценарий удаляется из списка.

## Режимы генерирования документов

Существуют следующие три режима генерирования документов:

- По расписанию;
- По команде от МРВ;
- По команде оператора.

Рассмотрим особенности настройки и выполнения сценариев в этих режимах.

## Генерирование документов по расписанию

### Ежедневно

## Периодически

## Каждую смену

Для этого режима в диалоге **Создание/редактирование сценария** можно выбрать один из трех вариантов задания расписания. Такими вариантами являются:

Ежедневно;

Периодически;

Каждую смену.

Рассмотрим особенности этих вариантов

### **Ежедневно**

Для первого варианта необходимо указать время и даты, когда надо осуществить подготовку документа. При настройке дат можно указать дни недели и числа месяца. В этом случае документ будет генерироваться в указанное время в перечисленные дни. Если не указать ни одной даты, то документ будет генерироваться каждый день в указанное время.

### **Периодически**

Этот вариант позволяет использовать те же настройки, что и для ежедневного варианта. Однако в этом случае есть возможность указать дополнительно период генерирования документа, например, один раз в час.

### **Каждую смену**

Для подготовки документа каждую смену надо выбрать соответствующий режим и создать файл описания смен. Этот файл называется **shifts.lst**. В нем надо перечислить времена начала всех смен. Каждая этого файла строка должна содержать время начала одной смены в следующем формате:

**ЧЧ:ММ**

где

**ЧЧ** - часы;

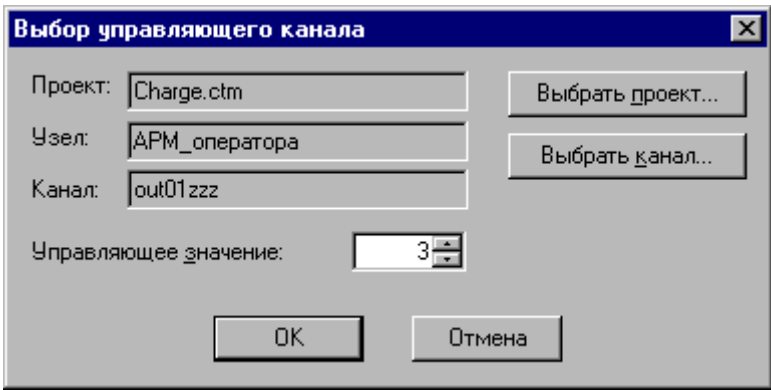
**ММ** - минуты.

Если файл **shifts.lst** отсутствует в рабочей директории, то считается, что предусмотрены две смены со временами начала 00:00 и 12:00.

### **Генерирование документов по команде МРВ**

Запуск выполнения сценария можно осуществлять по команде от МРВ. Для этого в разделе **Режим генерации** диалога **Создание/редактирование сценария** надо поставить

флаг **По команде**. После этого следует нажать ЛК на кнопке **Выбрать канал**. При этом на экран выводится следующий диалог.



Выбор управляющего канала

Проект: Charge.ctm      Выбрать проект...

Узел: АРМ\_оператора      Выбрать канал...

Канал: out01zzz

Управляющее значение: 3

OK      Отмена

В нем необходимо указать имя проекта, имя узла, с которого будет осуществляться управление и имя управляющего канала. Кроме того, в этом же диалоге надо задать значение управляющего канала, которое будет восприниматься сервером документирования как команда на выполнение сценария. Это значение обязательно должно быть отлично от 0.

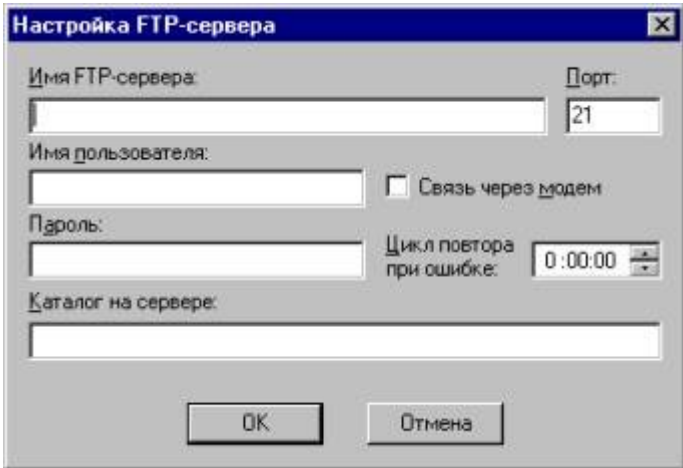
Если значение указанного канала равно заданному управляющему значению, сервер документирования выполнит данный сценарий и присвоит этому каналу значение 0.

#### **Оперативная генерация документа**

- Сервер документирования позволяет в случае необходимости осуществить внеочередное выполнение сценария. Для этого надо выполнить команду **Запустить сейчас** из меню **Работа** или нажать ЛК на соответствующей иконке инструментальной панели.

#### **Публикация документа на WEB-сервере**

Сервер документирования позволяет публиковать сгенерированный документ на WEB-сервере. Для этого при настройке сценария в разделе **Обработка** диалога **Создание/редактирование сценария** надо поставить флаг **Публиковать в Интернет**. После этого следует нажать ЛК на кнопке **Настройка сервера**. При этом на экран выводится следующий диалог.



Настройка FTP-сервера

Имя FTP-сервера:      Порт: 21

Имя пользователя:       Связь через модем

Пароль:      Цикл повтора при ошибке: 0:00:00

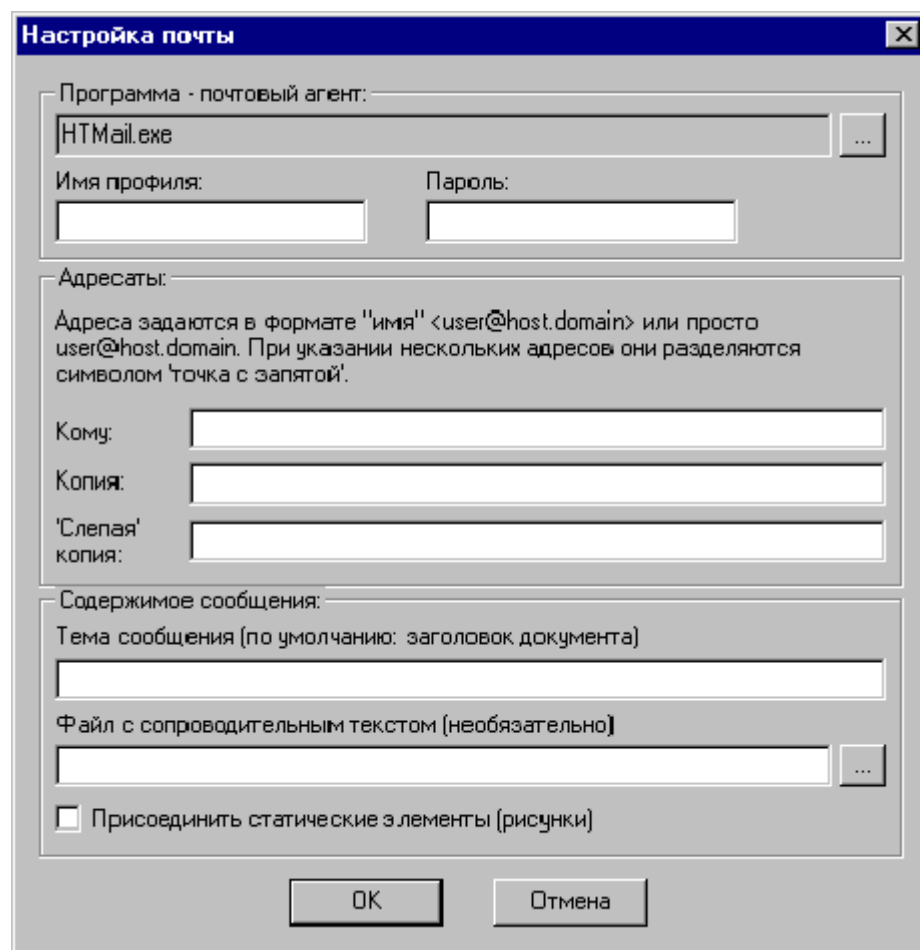
Каталог на сервере:

OK      Отмена

Здесь можно настроить все необходимые параметры, чтобы записать сгенерированный файл документа по **ftp:** на web-сервер.

### *Отправка документа по электронной почте*

Сервер документирования позволяет отправить сгенерированный документ по электронной почте. Для этого при настройке сценария в разделе **Обработка** диалога **Создание/редактирование сценария** надо поставить флаг **Отправить по эл. почте**. После этого следует нажать ЛК на кнопке **Настройка почты**. При этом на экран выводится следующий диалог.



В этом диалоге можно настроить все необходимые параметры для отправки сгенерированного файла документа по **e-mail**.

### *Вывод документа на принтер*

Для вывода генерируемого документа на принтер при настройке сценария в разделе **Обработка** диалога **Создание/редактирование сценария** надо поставить флаг **Вывести на печать**. В этом случае при выполнении сценария сгенерированный документ будет автоматически посылаться на принтер, установленный в системе по умолчанию.

- При необходимости осуществить внеочередное генерирование документа и вывод его на принтер сначала следует выбрать требуемый сценарий в списке. Затем надо выполнить команду **Напечатать сейчас** из меню **Работа** или нажать ЛК на соответствующей иконке инструментальной панели.



ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СР 14. [Документирование Trace Mode](#)  
**5.** Справочная система Trace Mode 5, раздел: «Документирование».

## 6. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ.

### Лабораторные работы в программе TRACE MODE 5

**ЛР 1:** Создание проекта, создание узла, автопостроение базы каналов контроллера, редактирование каналов;

**ЛР 2:** Тиражирование узлов проекта, автопостроение базы каналов операторской станции для обмена с другими узлами проекта и внешними контроллерами;

**ЛР 3:** Разработка и отладка программ управления на Техно FBD и Техно IL;

**ЛР 4:** Разработка графической базы для операторской станции;

**ЛР 5:** Организация архивирования.

**ЛР 6:** Организация документирования.

**ЛР 7:** Организация управления техпроцессом через Интернет.

В рамках первых трех ЛР мы создадим проект под названием БЫСТРЫЙ\_СТАРТ. В нем мы разработаем систему, которая отслеживает изменение давления и уровня жидкости в емкости.

В основе ЛР 4–7 лежат проекты БЫСТРЫЙ\_СТАРТ2 и ШИХТОПОДГОТОВКА, которые входят в комплект поставки инструментальной системы.

#### ЛР 1

Эта ЛР посвящена знакомству с редактором базы каналов. В нем будут рассмотрены следующие темы:

- создание проекта;
- создание узла проекта;
- автопостроение базы каналов для контроллера;
- редактирование базы каналов.

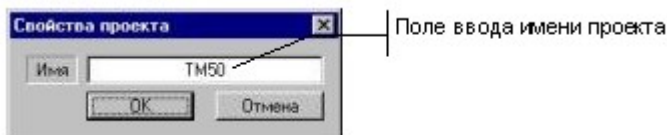
#### Создание проекта

Разработка любого проекта всегда начинается в редакторе базы каналов. Чтобы загрузить этот редактор, надо выполнить команду **Редактор базы каналов** из группы установки инструментальной системы в меню **Программы WINDOWS**.

При этом осуществляется запуск редактора базы каналов и на экране появляется его окно, показанное на рисунке.



Для создания нового проекта следует нажать ЛК на иконке инструментальной панели, обозначенной на предыдущем рисунке. При этом на экран выводится следующий диалог.



Имя проекту зададим **БЫСТРЫЙ\_СТАРТ**.

Проекты **БЫСТРЫЙ\_СТАРТ** и **БЫСТРЫЙ\_СТАРТ2** эмулируют работу контроллеров. Для задания имени проекта, содержащего физические контроллеры, нужно использовать латинские буквы.

Подтвердим завершение настройки параметров проекта нажатием ЛК на кнопке ОК. При этом диалог **Свойства проекта** исчезнет с экрана, а в заголовке окна редактора базы каналов и его строке статуса появится название нового проекта.

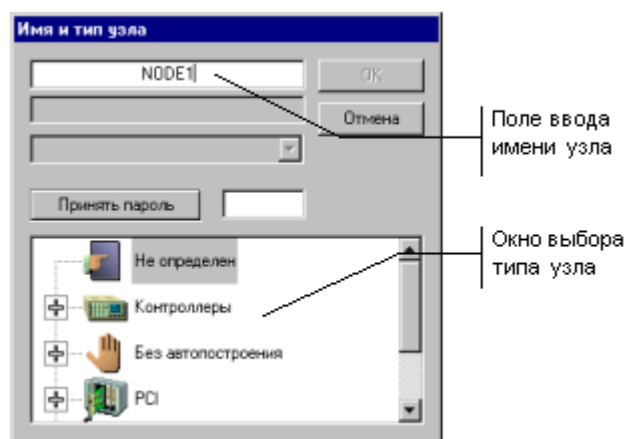
### Создание узлов проекта

Перейдем теперь к созданию структуры проекта. Она включает в себя перечень **узлов** – операторских станций и контроллеров, которые работают под управлением ТРЕЙС МОУД.

В нашем случае потребуется создать два узла. Один из них будет РС-контроллером, а второй – операторской станцией.

Для создания узла надо выполнить команду **Создать** из меню **Узел** или нажать ПК в рабочей области редактора базы каналов.

При этом на экране появится диалог **Имя и тип узла**, показанный на следующем рисунке.



Введем в соответствующем поле диалога название узла **КНТ1**. Это будет РС-контроллер. В окне выбора типа узла раскроем список узлов под названием **Контроллеры**. Для этого нажмем ЛК в области [+], расположенной левее названия. Для примера выберем тип контроллера MIC 2000 фирмы ADVANTECH.

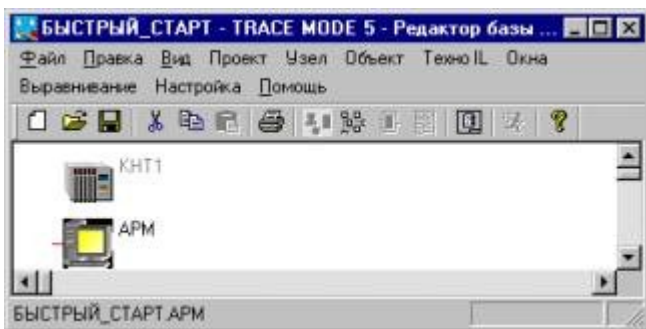
Подтвердим создание узла нажатием ЛК на кнопку ОК. При этом диалог **Имя и тип узла** исчезает с экрана, а в рабочем поле редактора базы каналов появляется обозначение созданного узла.



Повторим те же действия для создания нового узла, который будет операторской станцией. Имя этому узлу следует задать **АРМ** и выбрать следующий тип из класса **Большой**:



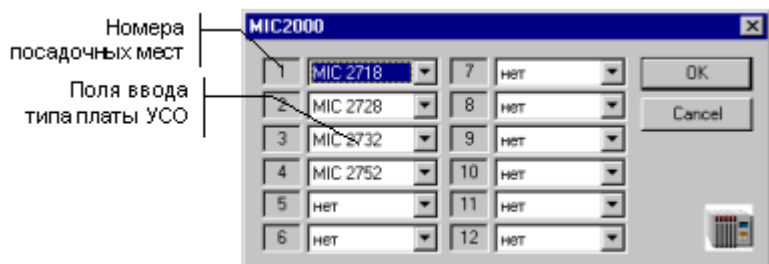
После выполнения этих действий окно редактора базы каналов будет иметь следующий вид.



### Автопостроение базы каналов для контроллера

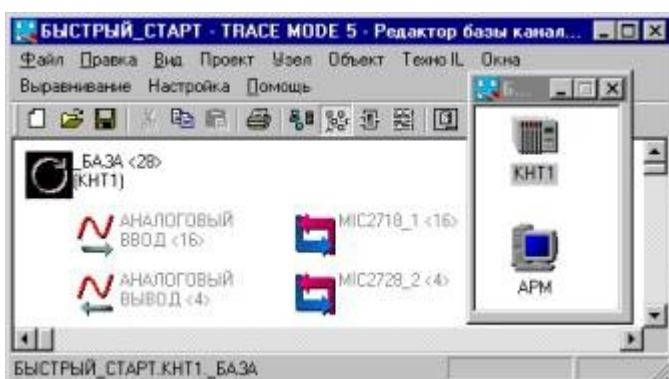
Перейдем теперь к созданию базы каналов контроллера. Для этого дважды нажмем ЛК на его изображении в рабочем поле окна редактора базы каналов. При этом экран будет выведен диалог ее настройки процедуры автопостроения. В нем для каждого слота контроллера можно указать тип используемой платы УСО. После этого автоматически настраивается обмен данными с этими платами.

Для контроллеров MIC2000 диалог настройки процедуры автопостроения выглядит следующим образом.



Укажем для первых четырех посадочных мест показанные на рисунке типы плат. Это платы аналогового ввода, аналогового вывода, дискретного ввода и дискретного вывода

соответственно. Нажмем ЛК на кнопке ОК. При этом открывается окно редактирования базы каналов, показанное на следующем рисунке.

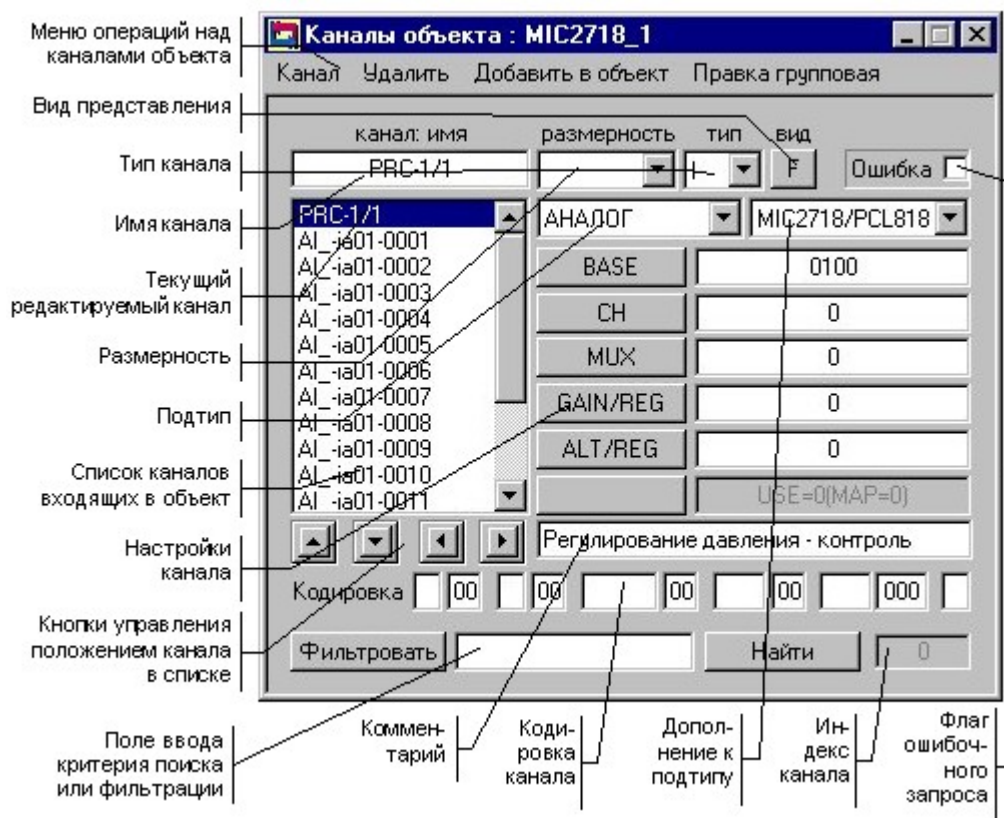


В этом окне выводятся изображения объектов базы каналов. Левая колонка – это стандартные объекты, которые заполняются каналами автоматически в соответствии с настройки последних.

Следующая колонка содержит объекты, созданные автопостроением для связи с платами УСО в контроллерах или с каналами другого узла проекта. Эти объекты имеют имена вида **NAME\_n**, где **NAME** – название типа или узла, а **n** – номер посадочного места. В нашем случае объект, расположенный в первом посадочном месте носит имя **MIC2718\_1<16>**. В скобках указано число каналов в объекте. Каналы из этих объектов присутствуют также в стандартных объектах типа **АНАЛОГОВЫЙ ВВОД**, **АНАЛОГОВЫЙ ВЫВОД**, **ДИСКРЕТНЫЙ ВВОД** и **ДИСКРЕТНЫЙ ВЫВОД**. Полный список каналов, присутствующих в редактируемой базе, доступен в стандартном объекте **БАЗА**.

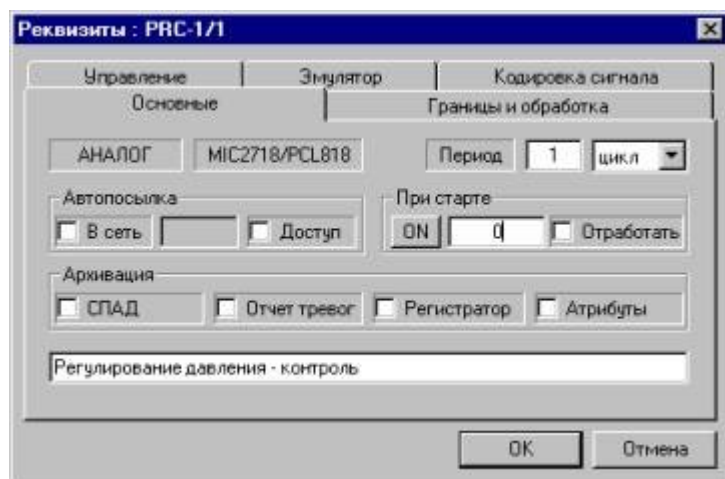
#### **Редактирование базы каналов**

Для редактирования каналов объекта **MIC2718\_1** дважды нажмем ЛК на его изображении. При этом на экране появится диалог **Каналы объекта**, показанный на следующем рисунке.



Выберем первый канал в списке данного объекта и в поле ввода имени канала введем для него новое имя (**PRC-1/1**), а в поле ввода комментария следующий текст: **Регулирование давления – контроль**. Далее установим размерность канала – **ати**.

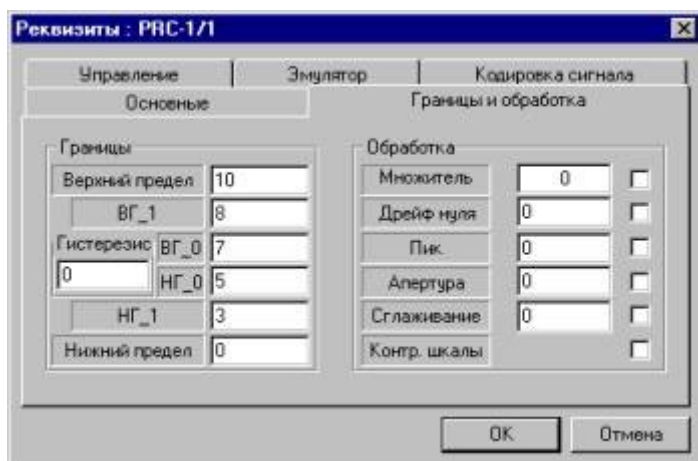
Для дальнейшей настройки канала следует дважды нажать ЛК на его имени в списке. При этом на экран будет выведен диалог **Реквизиты**. Вид этого диалога показан на следующем рисунке.



Этот диалог позволяет ввести частоту и фазу пересчета канала, настроить и отладить первичную обработку сигнала в канале, ввести шкалу и аварийные границы канала, настроить вызов программ для более сложной обработки сигнала или управления, а также настроить ряд других параметров канала.

Нажатием ЛК установим в этом бланке диалога флаг **Доступ**. Этот флаг используется при автопостроении баз каналов операторских станций, запрашивающих данные у этого узла. Он указывает каналы, которые следует опрашивать.

Перейдем на бланк **Границы и обработка** данного диалога и введем значения границ шкалы и аварийных границ, как показано на рисунке ниже. Завершим редактирование канала нажатием ЛК на кнопке ОК диалога **Реквизиты**. Далее закроем диалог **Каналы объекта** для объекта MIC2718\_1.



Теперь войдем в редактирование каналов объекта MIC2728\_2, который содержит каналы, связанные с платой УСО второго посадочного места. В нем установлена плата аналогового вывода.

Выберем первый в списке канал. Установим для него имя **PRC-1/2**, комментарий – **Регулирование давления – управление** и размерность - %.

Далее войдем в диалог **Реквизиты** для канала **PRC-1/2** и установим флаг **Доступ**.

Разработка программы PID-регулятора и привязка ее к каналам будет рассмотрена в ЛР 3.

Теперь войдем в меню редактора **Файл** и выполним команду **Сохранить**. После этого выйдем из редактора базы каналов с помощью команды **Выход** того же меню.

На этом ЛР 1 закончена.


## ЛР 2

Эта ЛР продолжает знакомство с работой в редакторе базы каналов. В нем будут рассмотрены следующие темы:

- тиражирование узлов проекта;
- автопостроение базы каналов операторской станции для обмена данными с другими узлами проекта;
- автопостроение базы каналов операторской станции для обмена данными с внешними контроллерами.


## Тиражирование узлов проекта

Запустим редактор базы каналов, как это было описано в ЛРе 1, и загрузим в него проект **БЫСТРЫЙ\_СТАРТ**. Для загрузки надо выполнить одну из следующих операций:


- команда **Открыть** из меню **Файл**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **CTRL-O**.

При этом на экран выводится диалог выбора файла проекта. В нем выводится список файлов проектов из рабочей директории ТРЕЙС МОУД. Выберем файл **БЫСТРЫЙ\_СТАРТ**. После этого указанный проект будет загружен в редактор базы каналов.

Предположим, что автоматизируемый технологический объект имеет два однотипных участка, каждый из которых управляется отдельным контроллером. Соответственно, оба эти контроллера должны иметь одинаковые конфигурации и программы управления. В предыдущей ЛР мы создали базу каналов для одного контроллера. Этот узел имеет имя **КНТ1**. Создадим теперь новый узел в проекте путем тиражирования узла **КНТ1**. Для этого его нужно выделить нажатием ЛК и выполнить одно из следующих действий:

- выполнить команду **Копировать** из меню **Правка**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **CTRL-C**.

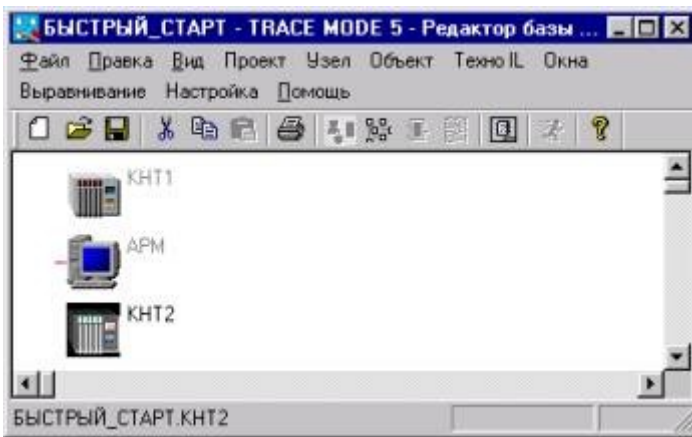
Затем, чтобы вставить содержимое буфера в проект, надо выполнить одно из следующих действий:

- выполнить команду **Вставить** из меню **Правка**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **CTRL-V**.

После этого в рабочем поле редактора базы каналов появится новый узел с тем же графическим идентификатором, что и у **КНТ1**. Его имя образовано следующим образом: **CNODEn**, где **n** – сетевой номер данного узла.

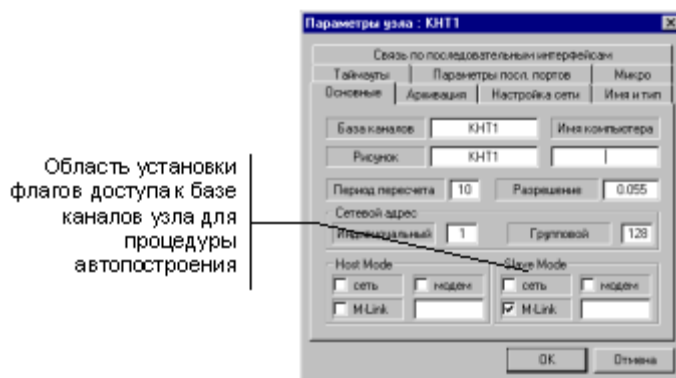
Изменим имя нового узла. Для этого нажмем ПК на его изображении в рабочей области редактора. При этом на экран выводится диалог **Параметры узла**. Откроем в нем бланк **Имя и тип** и в соответствующем поле введем имя **КНТ2**. После этих операций окно редактора базы каналов будет выглядеть следующим образом.





### Автопостроение базы каналов для обмена данными с другими узлами проекта

При автопостроении каналов для связи с другими узлами проекта существенно, чтобы эти узлы поддерживали те же линии обмена данными (локальная сеть, последовательный интерфейс или коммутируемые телефонные линии). Поддержка линии передачи данных настраивается в диалоге **Параметры узла**, для входа в который следует нажать ПК на изображении узла.



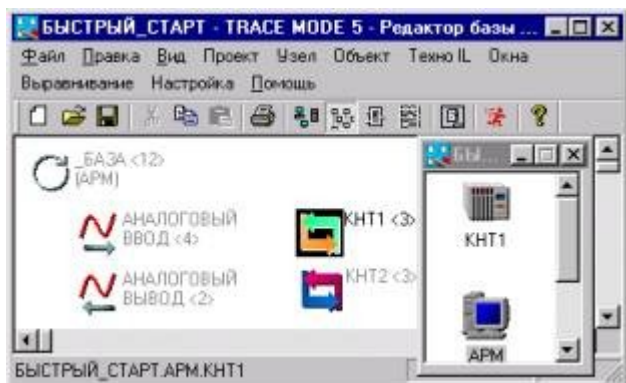
На рисунке обозначена область установки флага доступа к базе каналов при выполнении процедуры автопостроения. В нашем случае для узлов **KHT1** и **KHT2** должен быть установлен флаг **M-Link**, что означает связь по последовательному интерфейсу.

Первый переход к редактированию базы каналов любой операторской станции сопровождается выводом на экран диалог настройки автопостроения обмена данными с другими узлами проекта. Это могут быть как операторские станции, так и контроллеры. Следующий рисунок демонстрирует вид этого диалога.



Для вызова этого диалога повторно следует использовать команду **Автопостроить** из меню **Узел**.

В поле 1 зададим связь с контроллером **КНТ1**, а в поле 2 – с **КНТ2**. Подтвердим настройки нажатием ЛК на кнопке ОК. При этом в рабочей области редактора появится база каналов узла **АРМ**, как показано на следующем рисунке.



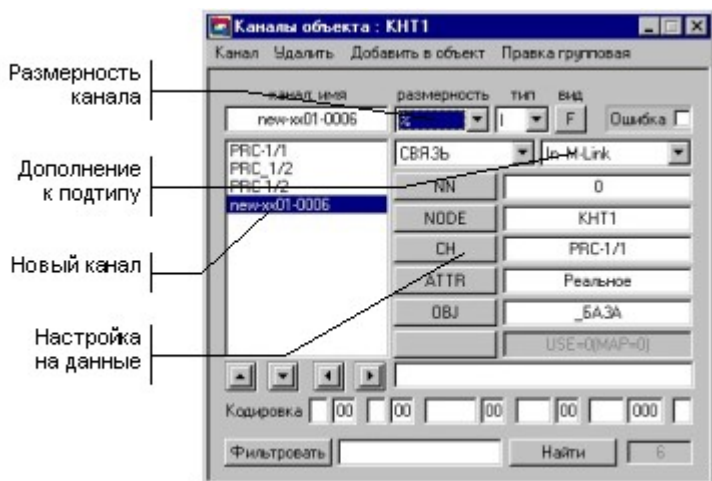
Здесь присутствуют два объекта, имена которых образованы из имен соответствующих узлов. В них создаются каналы, запрашивающие значения соответствующих каналов этих узлов.

Выведем на экран диалог каналов объекта **КНТ1**. Для этого дважды нажмем ЛК на его изображении. Здесь присутствуют три канала. Два имеют те же имена, комментарии, размерности, шкалы и границы, что и соответствующие каналы в узле КНТ1. Третий канал – выходной. Он имеет тип О. Такие каналы создаются для обмена с управляющими каналами другого узла, не имеющих законов управления. При необходимости их можно удалить вручную.

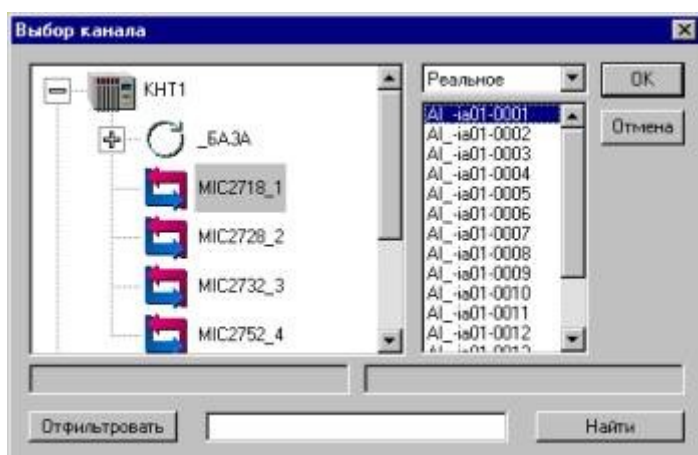
В базе каналов АРМ воспроизвелись только те каналы, для которых были установлены флаги доступа

### Создание и настройка каналов

Откроем диалог **Каналы объекта** для объекта **КНТ1**. Для этого дважды нажмем ЛК на его изображении. Выделим в списке канал **PRC-1/1** и выполним команду **Создать по образцу** из меню **Канал**. При этом в списке появился новый канал. Он будет иметь те же настройки, что и **PRC-1/1**. Изменим его размерность, как показано на следующем рисунке.



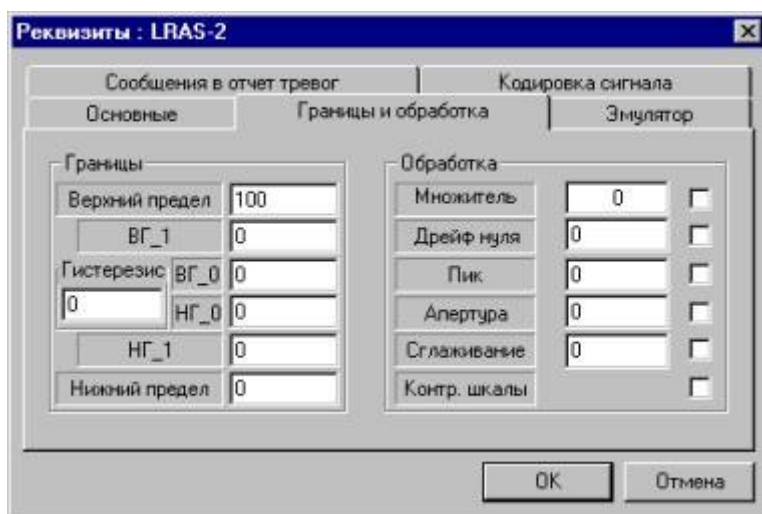
Изменим источник данных для нового канала. Для этого нажмем ЛК на настройке **СН**. На экране появится следующий диалог.



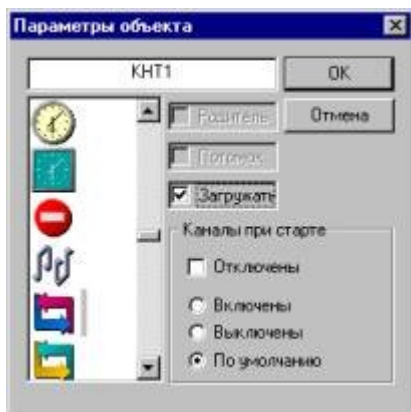
В нем указывается узел проекта, объект базы каналов этого узла, канал в выбранном объекте и его атрибут для копирования в запрашивающий канал.

В нашем случае узел оставим без изменения, выберем объект **АНАЛОГОВЫЙ ВВОД** и канал **AI\_ia01-0001** и нажмем ЛК на кнопке **OK**. После этого изменим имя канала на **LRAS-2** и введем комментарий **Контроль уровня**.

После этого войдем в диалог **Реквизиты** нового канала, дважды нажав ЛК на его имени, и зададим для него границы шкалы 0 и 100.




Для дальнейшего доступа к объекту **КНТ1** из редактора представления данный нажмем ПК на его изображении и в появившемся диалоге **Параметры объекта** установим флаг **Загружать**.



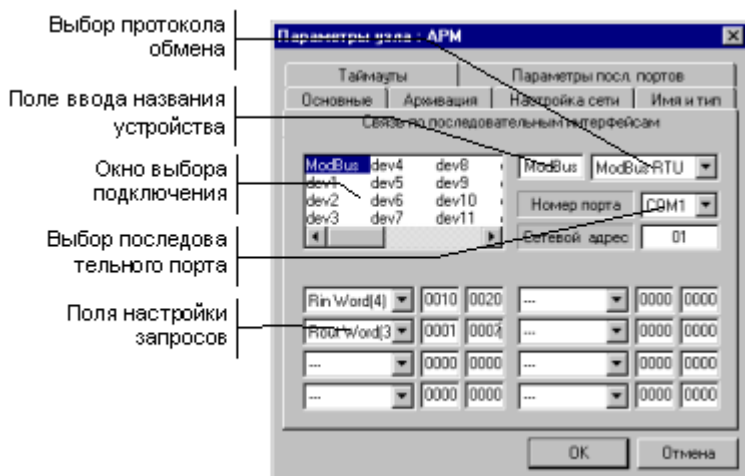
Прделаем все аналогичные действия для объекта **КНТ2**.

### Автопостроение базы каналов для обмена данными с внешними контроллерами

Подключим к узлу АРМ контроллер типа **MODICON**. Для этого перейдем в окно структуры проекта. Это можно сделать одним из следующих способов:

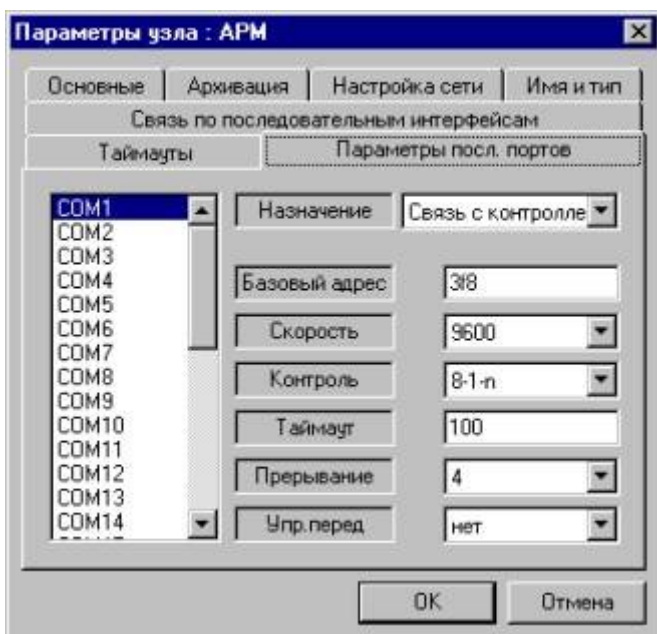
- выполнить команду **Узлы** из меню **Окна**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **ALT-1**.

Войдем в диалог **Параметры узла** операторской станции АРМ. Для этого нажмем ПК на ее изображении. Откроем бланк **Связь по последовательным интерфейсам** данного диалога.

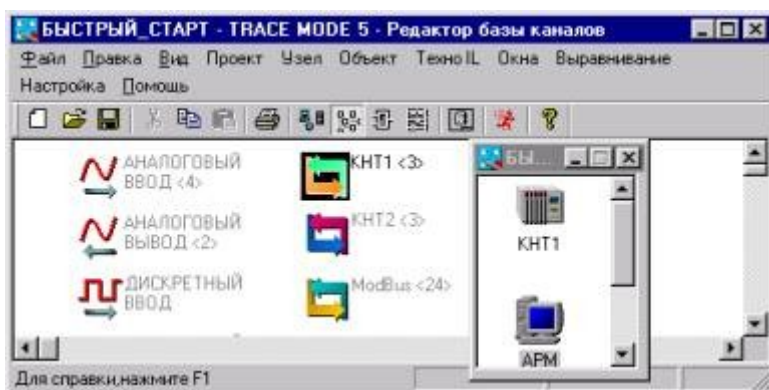


Выберем в окне подключения устройство **dev0** и в поле ввода названия устройства введем **ModBus**. Выберем протокол **ModBus RTU** и назначим порт для связи **COM1**. После этого опишем переменные, которые надо запрашивать с контроллера. На рисунке показан вид бланка после завершения всех необходимых настроек.

После того как связь описана и указаны переменные, которые надо запрашивать, нужно осуществить настроить последовательный порт. Это реализуется в бланке **Параметры посл. портов** того же диалога. Вид этого бланка показан на следующем рисунке.



Теперь после входа в окно редактирования базы каналов узла **АРМ** в нем будет присутствовать новый объект **ModBus**. В нем находятся каналы, настроенные на запрос указанных переменных.



**Внимание!** Пользователям ТРЕЙС МОУД на 128 точек ввода/вывода после автопостроения необходимо изменить тип узла АРМ на **Малый** из раздела **Без автопостроения**. Для этого следует использовать бланк **Имя и тип** диалога **Параметры узла**. Без выполнения этой процедуры нельзя создать графическую базу узла АРМ в редакторе представления данных.

### ЛР 3

В этой ЛР разбираются особенности разработки управляющих программ в ТРЕЙС МОУД. Здесь будут рассмотрены встроенные в систему языки программирования и разобраны примеры разработки и отладки программ.

Эта ЛР включает в себя следующие темы:

- создание и разработка FBD-программы;
- подключение FBD-программы к каналам;
- отладка FBD-программы;


- создание, разработка и подключение к системе ПЛ-программы.

### Создание FBD-программы

В этом разделе мы создадим FBD-программу, реализующую PID-регулятор. В ней будет вычисляться рассогласование параметра и задания, формироваться величина управляющего воздействия по PID-закону с ограничением по заданным границам.

Запустим редактор базы каналов и загрузим в него проект БЫСТРЫЙ\_СТАРТ. Описание этих операций было приведено в предыдущей ЛР.

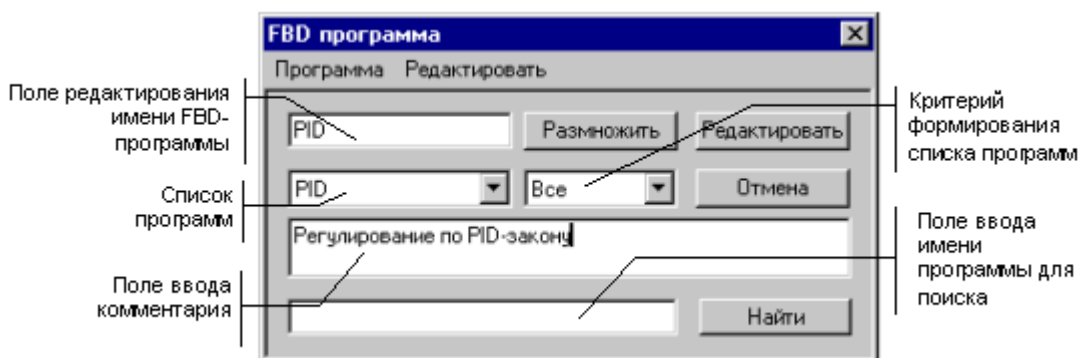
Чтобы создать FBD-программу, следует сначала указать, на каком узле она будет использоваться. Для этого нужно либо войти в режим редактирования базы каналов этого узла, либо просто выделить его в окне структуры проекта. В нашем случае выделим узел **КНТ1**. Далее для перехода в окно редактирования FBD-программ надо выполнить одно из следующих действий:

- выполнить команду **FBD-программы** из меню **Окна**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **ALT-3**.

При этом на экране появляется диалог **FBD-программа**. В нем можно выбрать FBD-программу для редактирования или создать новую.

Для создания новой программы надо выполнить команду **Создать** из меню **Программа** диалога **FBD-программа**. По умолчанию создаваемой программе присваивается имя **FormN**, где **N** – ее номер по порядку в данном узле. Для изменения имени программы в данном диалоге предусмотрено специальное поле.

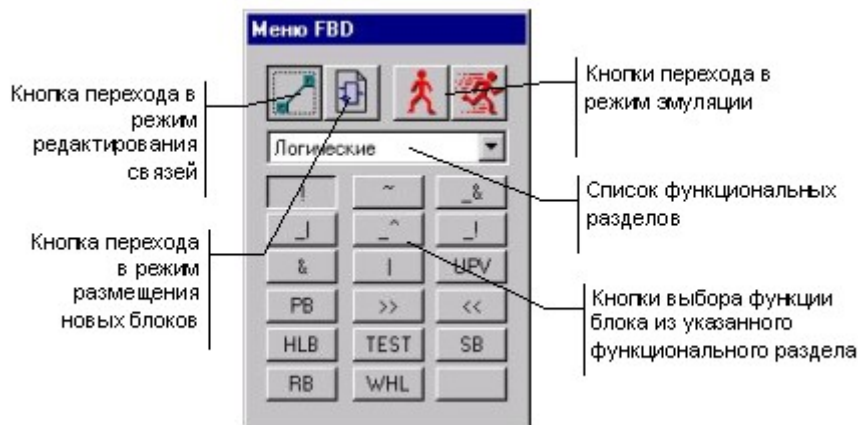
Создадим новую программу, Присвоим ей имя PID и введем для нее комментарий, как показано на следующем рисунке.



Для перехода к редактированию созданной FBD-программы следует нажать ЛК на кнопке **Редактировать**. При этом диалог **FBD-программа** исчезнет с экрана, а в рабочее поле редактора базы каналов будет выведена выбранная FBD-программа. Кроме того, на экране появится диалог управления редактированием **Меню FBD**.



В нашем случае, поскольку программа только что создана, рабочее поле редактора будет пустым. В нем будет только присутствовать диалог управления редактированием **Меню FBD**, показанный на следующем рисунке:

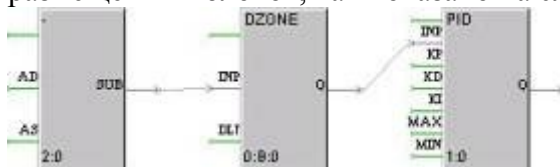


Нажатием соответствующей кнопки в диалоге **Меню FBD** перейдем в режим размещения новых блоков. Далее следует выполнить следующую последовательность действий:

- выбрать функциональный раздел **Арифметические**;
- выбрать блок вычитания (-) и разместить его нажатием ЛК в рабочем поле;
- выбрать функциональный раздел **Управление**;
- выбрать блок зоны нечувствительности (**DZONE**) и разместить его в рабочем поле;
- выбрать функциональный раздел **Регулирование**;
- выбрать блок **PID** и разместить его в рабочем поле.

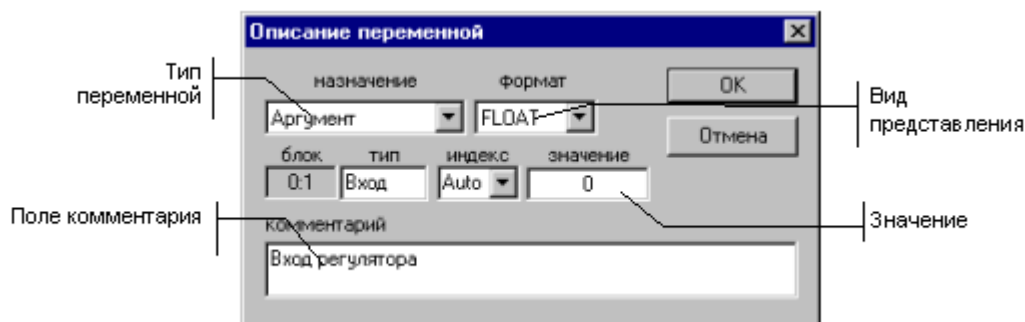


Далее следует перейти в режим редактирования связей. Это делается нажатием ЛК на соответствующей кнопке диалога **Меню FBD**. Теперь следует связать входы и выходы размещенных блоков, как показано на следующем рисунке.



Для соединения входа одного блока с выходом другого надо сначала выделить с помощью ЛК этот вход, затем снова нажать ЛК (при этом будет слышен звуковой сигнал) и, удерживая ее, переместить курсор в область второго конца связи. Для уничтожения связи следует выделить ее со стороны входа блока и нажать клавишу **DEL**.


Далее следует описать переменные и константы данной программы. Для описания любой переменной программы следует дважды нажать ЛК на соответствующем входе или выходе функционального блока. При этом на экране появится следующий диалог.



Настроим входы и выходы блоков следующим образом.

Блок	Вход, выход	Тип	Комментарий
(-)	AD	аргумент	Задание регулятора
	AS	аргумент	Вход регулятора
DZONE	DLT	константа	Зона нечувствительности
PID	KP	константа	Коэффициент при пропорциональной составляющей
	KD	константа	Коэффициент при дифференциальной составляющей
	KI	константа	Коэффициент при интегральной составляющей
	MIN	константа	Минимум управления
	MAX	константа	Максимум управления
	Q	аргумент	Выход регулятора

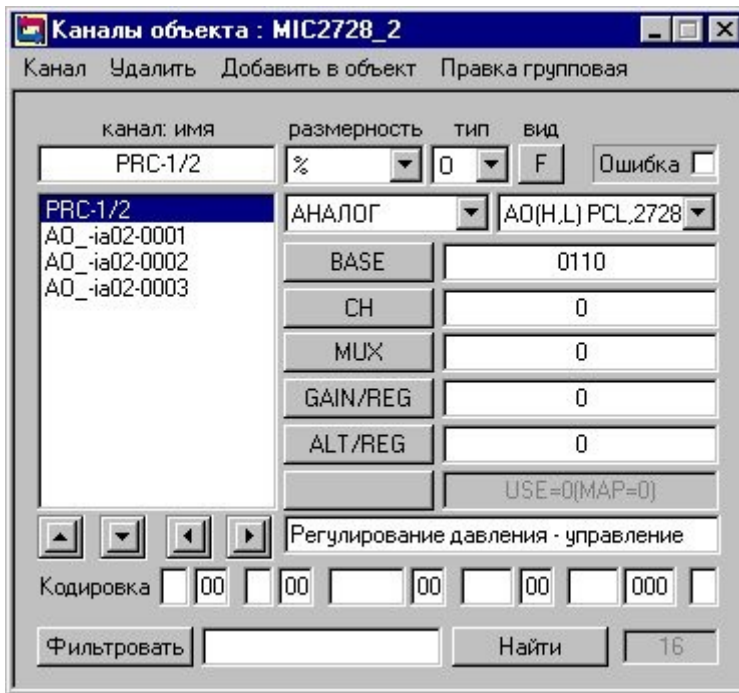
На этом разработка программы завершена. Теперь следует перейти в окно редактирования базы каналов узла **КНТ1** для ее привязки к каналам. Это реализуется одним из следующих способов:

- выполнить команду **Объекты** из меню **Окна**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **ALT-2**.

### Подключение FBD-программы к каналам

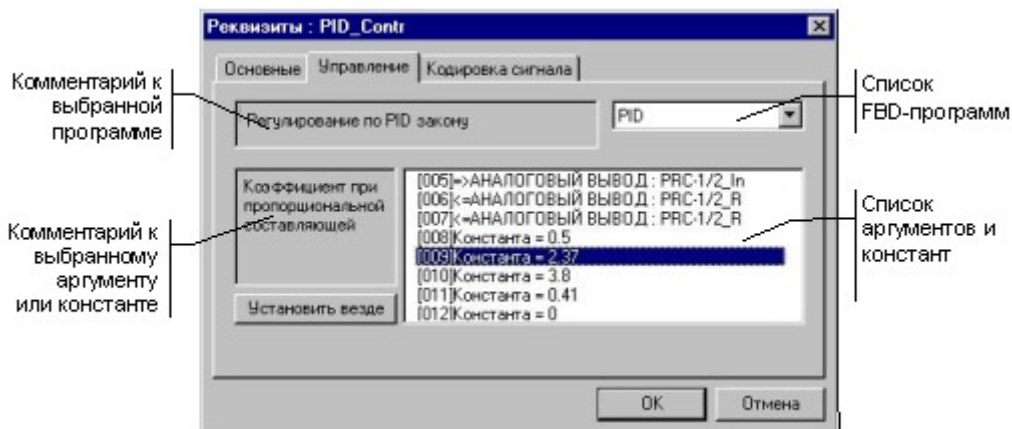
Создадим специальный канал **PID\_Contr** в узле **КНТ1** и подключим к нему FBD-программу **PID**. Для создания канала войдем в диалог **Каналы объекта** для объекта **MIC2728\_2**.





Выделим в списке канал **PRC-1/2** и выполним команду **Создать по образцу** из меню **Канал**. При этом в списке появится новый канал. Зададим ему имя **PID\_Contr** в поле **Канал, имя** и подтип **Управление** в поле **Подтип**.

Далее свяжем FBD-программу **PID** с каналом **PID\_Contr**. Для этого откроем диалог **Реквизиты** канала **PID\_Contr**, дважды нажав ЛК на имени канала в списке. Войдем в бланк **Управление** диалога **Реквизиты** и в поле выбора FBD-программ укажем **PID**, как показано на следующем рисунке.

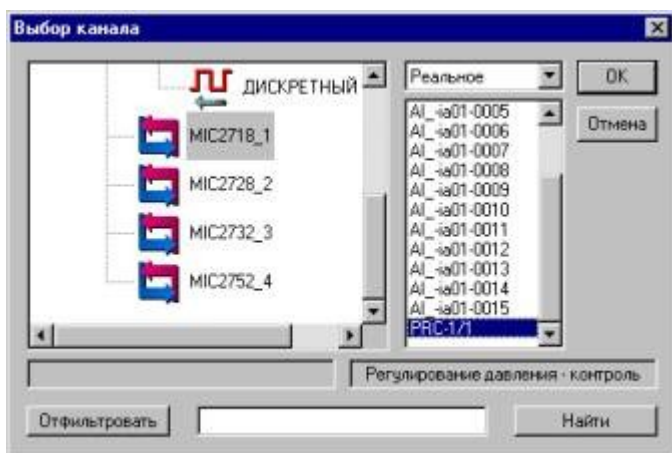


В этом бланке присутствует список для выбора программы, поле комментария к программе, список ее аргументов и констант, а также комментарий к настраиваемому элементу этого списка.

Для настройки любого аргумента или константы надо дважды нажать ЛК на соответствующей строке списка. Если выбрана константа, то на экране появится диалог **Значение переменной**. В нем можно ввести значение константы для данного вызова программы. Вид этого диалога показан на рисунке.



При настройке аргумента на экран выводится следующий диалог.



В нем выбирается канал и его атрибут, со значением которого связывается настраиваемый аргумент. Свяжем вход регулятора с реальным значением канала PRC-1/1 объекта MIC2718\_1, выход – с входом канала PRC-1/2 объекта MIC2728\_2, а задание регулятора - с границей ВГ\_1 канала PRC-1/2. Завершим настройку нажатием ЛК на кнопке ОК диалогов **Реквизиты**.

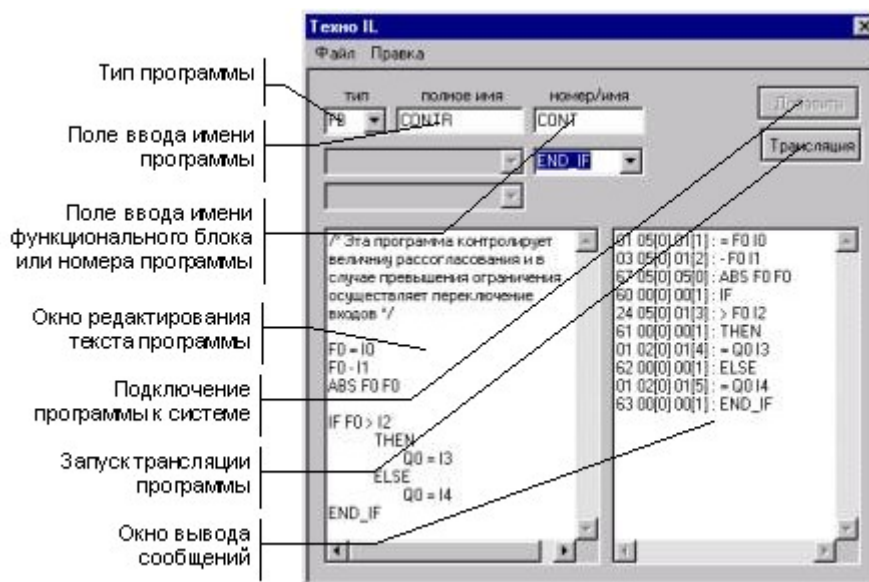
На этом настройка FBD-программы на каналы завершена.

### Создание ПЛ-программы

В Техно FBD существует более 150 стандартных блоков, реализующих различные функции. Используя язык Техно ПЛ можно создать собственный функциональный блок и подключить его к системе. Кроме того, Техно ПЛ позволяет разрабатывать метапрограммы, запускаемые параллельно с пересчетом базы каналов.

В рамках данной ЛР мы разработаем функциональный блок, который контролирует рассогласование значений двух первых входов и коммутировать на выход значение одного из входов по условию превышения рассогласованием значения третьего входа.

Чтобы создать программу на Техно ПЛ, надо войти в окно редактирования базы каналов и выполнить команду **Создать** из меню **Техно ПЛ** редактора базы каналов. При этом на экране появится следующий диалог.



Синтаксис языка Техно IL подробно описан в разделе, посвященном встроенным языкам разработки алгоритмов.

Оставим тип заданным по умолчанию – **FB** (функциональный блок). Присвоим программе имя **CONTR**, а создаваемому функциональному блоку – **CONT**. Далее введем в левом окне текст программы, как показано на рисунке выше.

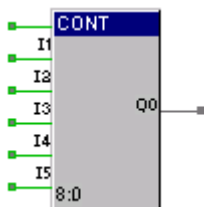
Чтобы проверить синтаксис программы, выполним команду **Дамп** из меню **Правка**. Результат проверки выводится в правом окне.

Если сообщений об ошибках нет, то программу можно подключать к системе. Но сначала ее надо транслировать. Это реализуется нажатием ЛК на кнопке **Трансляция**. Если трансляция прошла успешно, то в правом окне появится надпись **ОК**. Теперь для подключения программы надо нажать ЛК на кнопке **Добавить**.

Если теперь перейти в окно редактирования FBD-программ и в диалоге **Меню FBD** выбрать раздел **Техно IL\_1**, то в нем будет присутствовать только что созданный блок. Вид данного меню показан на следующем рисунке.



При размещении данного блока в рабочей области он будет выглядеть следующим образом.



## ЛР 4

Эта ЛР посвящена знакомству с редактором представления данных. Здесь будут рассмотрены основные приемы разработки графического интерфейса операторских станций.

Эта ЛР включает в себя следующие темы:

- создание графической базы узла;
- создание статического рисунка;
- отображение в графическом виде значений каналов;
- тиражирование графических фрагментов;
- эмуляция работы графической базы.

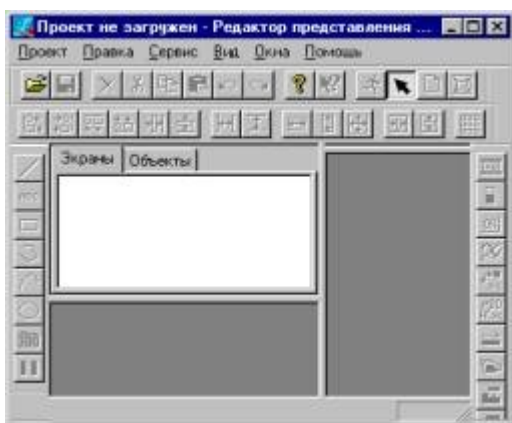
### *Создание графической базы узла*

Разработка графического интерфейса для операторских станций осуществляется в редакторе представления данных. В него загружается структура проекта, созданная в редакторе базы каналов.

Выбрав требуемый узел проекта, можно редактировать его графическую базу. Эта база включает в себя все графические фрагменты, которые выводятся на монитор данной операторской станции.

### **Запуск редактора представления данных**

Для запуска редактора представления данных следует войти в папку **Программы** главного меню WINDOWS, затем - в папку **Trace Mode 5** и выполнить команду **Редактор представления данных**. При этом на экране появится окно редактора. Его вид показан на следующем рисунке.




Редактор представления данных имеет главное меню, рабочую область, строку статуса, навигатор проекта и четыре инструментальные панели. Их расположение при первой загрузке редактора показано на рисунке. Оно не является жестким и может настраиваться.

В навигаторе проекта выводятся список узлов проекта, состав их графических баз и списки загруженных графических библиотек. Инструментальные панели используются для выбора графических элементов и управления редактированием графической базы.

Рабочая область редактора при загрузке не содержит никаких изображений, а инструментальные панели недоступны. Для создания и редактирования графических экранов необходимо загрузить проект и загрузить или создать графическую базу любого узла проекта.

### Загрузка проекта и создание графической базы

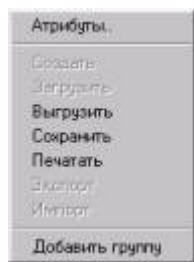
Загрузить проект в редактор представления данных можно одним из следующих способов:

- выполнить команду **Открыть** из меню **Проект**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **CTRL-O**.

При этом на экран выводится диалог выбора файла структуры проекта. Выберем проект **БЫСТРЫЙ\_СТАРТ2**. После загрузки проекта в бланке **Экраны** навигатора проекта появляется список присутствующих в проекте узлов: **ARM**, **PLC1** и **PLC2**.

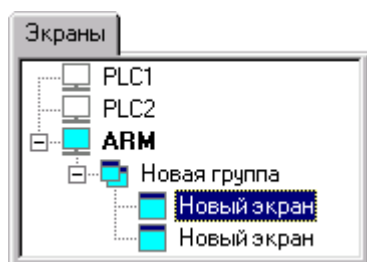
Проект **БЫСТРЫЙ\_СТАРТ2** эмулирует работу контроллеров. Для задания имени проекта, содержащего физические контроллеры, нужно использовать латинские буквы.

Выберем нажатием ЛК на бланке **Экраны** навигатора проекта операторскую станцию **ARM**. Затем нажатием ПК войдем в меню узлов этого бланка, показанное на следующем рисунке.

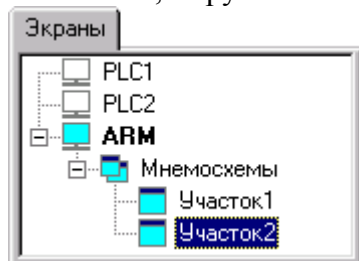


Выполним команду **Добавить группу**. При этом для узла **ARM** создастся группа экранов. Ей автоматически присваивается имя **Новая группа**. Далее выделим эту группу и нажмем ПК. При этом на экран выводится меню групп.

Выполним в этом меню команду **Добавить экран**. Снова войдем в это же меню и выполним ту же команду. После этого бланк **Экраны** навигатора проекта будет выглядеть следующим образом.

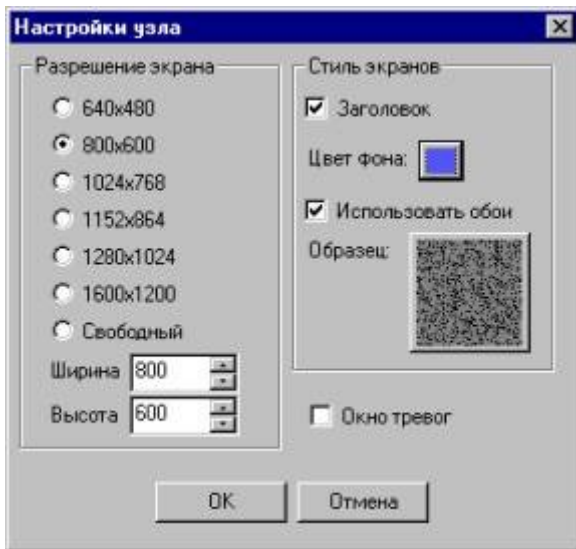


Изменим имена экранов и группы. Для этого надо выделить их нажатием ЛК на имени и после этого нажать ЛК повторно. Первому экрану дадим имя **Участок 1**, второму - **Участок 2**, а группе - **Мнемосхемы**. Бланк **Экраны** примет теперь следующий вид.

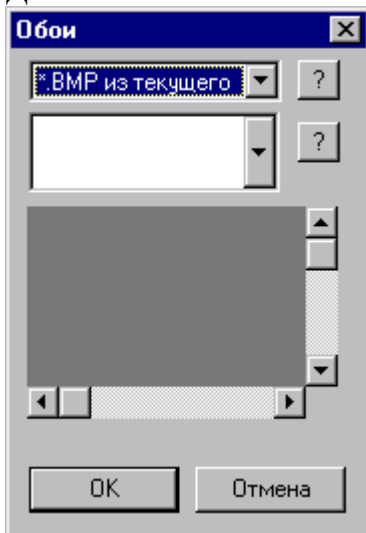


### Настройка атрибутов экранов

Настроим оформление экранов, которое включает в себя разрешение, наличие заголовка, цвет фона или обои. Для этого надо нажатием ЛК выделить узел **ARM** в бланке **Экраны** навигатора проекта. Затем нажатием ПК на имени узла войдем в меню узлов и выполним команду **Атрибуты**. При этом на экран выводится следующий диалог.

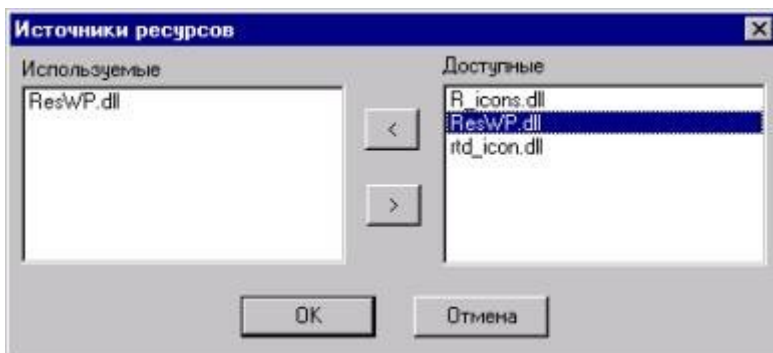


Зададим разрешение экрана 800x600 и поставим флаги наличия заголовка и использования обоев, как показано на рисунке. Далее следует выбрать текстуру для обоев. Для этого нажмем ЛК на кнопке **Образец**. На экране появится следующий диалог.




Чтобы воспользоваться текстурами, поставляемыми с системой, следует подключить их в качестве ресурсов. Для этого надо нажать ЛК на кнопке с символом "?" рядом с верхним списком.

При этом на экран выводится диалог подключения ресурсов.



Текстуры хранятся в библиотеке **ResWP.dll**. Перенесем эту библиотеку из окна

**Доступные** в окно **Используемые** с помощью кнопки , как показано на рисунке. Подтвердим подключение библиотеки нажатием ЛК на кнопке **ОК**. Далее в диалоге **Обои** в первом списке в качестве источника укажем только что подключенную библиотеку.

При этом становится доступным нижний список, где выводится содержимое библиотеки. Выберем из библиотеки понравившуюся текстуру и нажмем ЛК на кнопке **ОК**.

Сохраним созданную графическую базу. Для этого выполним команду **Сохранить** из меню **Проект**.

### ***Разработка графического интерфейса***

Разработка графического интерфейса заключается в размещении на экранах графической базы статических элементов рисования и динамических форм отображения. Графические элементы выбираются с помощью соответствующих инструментальных панелей. При этом на экран выводится диалог настройки их атрибутов (цвет фона, привязка к каналам и пр.).

#### **Создание статического рисунка**

[Рисование объемных элементов](#)

[Редактирование рисунка](#)

[Рисование рамок](#)

[Размещение статического текста](#)

Двойным нажатием ЛК на имени экрана **Участок 1** в бланке **Экраны** выведем его в рабочее поле редактора. Ниже показан статичный рисунок, который надо создать для этого экрана.



Рассмотрим последовательность его создания.

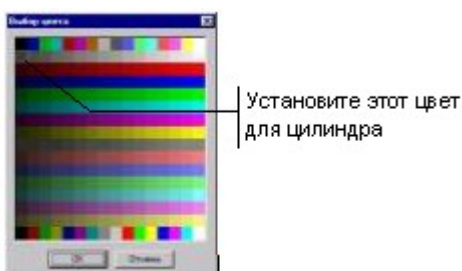
#### **Рисование объемных элементов**



Сначала разместим на экране объемный элемент – вертикальный цилиндр. Для этого выберем в панели элементов рисования соответствующую группу, как показано на следующем рисунке.

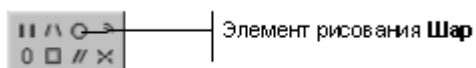


При настройке цвета следует указать самый темный цвет из светло-серой гаммы, как показано на следующем рисунке.

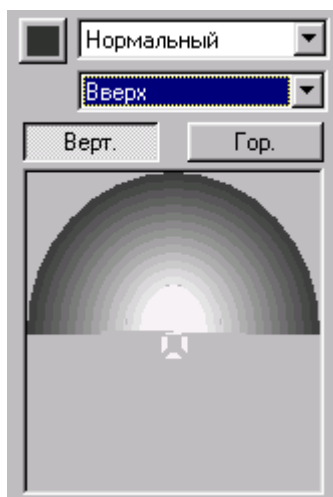


Далее разместим цилиндр на экране **Участок 1**. Для этого переведем курсор мыши в рабочую область редактора, выберем место, где должен располагаться нижний левый угол элемента и нажмем ЛК – это будет точка привязки цилиндра. После этого на экране появляется контурный прямоугольник, размеры которого меняются при перемещении мыши. Установим нужный размер цилиндра и нажмем ЛК повторно. Контурный прямоугольник исчезнет с экрана, а вместо него появится цилиндр такого же размера.

Теперь пририсует к верхнему торцу цилиндра эллиптическую крышку. Для этого нажмем ЛК еще раз на иконке объемных элементов и в появившемся меню выберем элемент **Шар**.




Настроим атрибуты элемента рисования **Шар**, как показано на следующем рисунке.

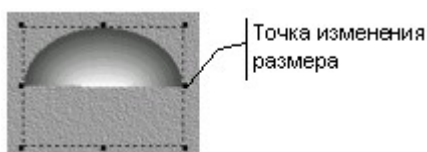


Порядок размещения аналогичен размещению цилиндра.

## Редактирование рисунка

 Чтобы изменить размер или положение графического элемента следует перейти в режим редактирования. Для этого нажмем ЛК на специальной иконке системной инструментальной панели.

Переведем курсор в рабочую область экрана, подведем его к границе графического элемента и при изменении его формы нажмем ЛК. При этом графический элемент обводится контурным прямоугольником, как показано на рисунке.



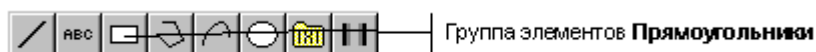
Для изменения размеров графического элемента надо перевести курсор мыши в область контурного прямоугольника, к одной из 8 точек изменения размера. При этом появятся стрелки, показывающие направление изменения. Далее следует нажать ЛК и, удерживая ее в нажатом состоянии, перемещением мыши добиться требуемого размера графического элемента. После того как ЛК будет отпущена, новый размер графического элемента зафиксируется.

Чтобы переместить графический элемент в другую область экрана, надо подвести курсор к контурному прямоугольнику. Однако курсор не должен попадать на точки изменения размера. При смене вида курсора следует нажать ЛК и, не отпуская ее, перемещением мыши изменить положение графического элемента. После того как ЛК будет отпущена, новое положение элемента зафиксируется.

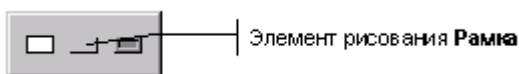
## Рисование рамок

Нарисуем теперь рамку по центру емкости. В ней затем будет размещена гистограмма, отображающая уровень. Вторую рамку разместим под емкостью для вывода в ней числовых значений давления и уровня.

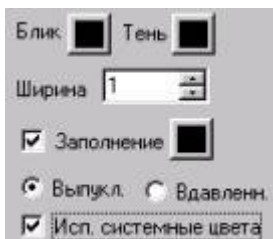
Нажмем дважды ЛК на иконке группы прямоугольников инструментальной панели элементов рисования, показанной на следующем рисунке.



При этом на экране появится меню выбора элемента группы. Нажмем ЛК на элементе **Рамка**, показанном на следующем рисунке.



После этого пиктограмма выбранного элемента появится в соответствующей ячейке инструментальной панели, а на экран будет выведен диалог настройки атрибутов элемента рисования **Рамка**. Вид этого диалога показан на следующем рисунке.

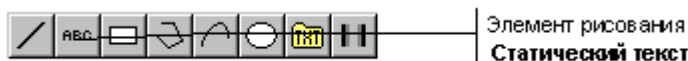


Установим в этом диалоге два флага: наличие заполнения и использование системных цветов WINDOWS для объемных элементов. После этого разместим две рамки, как показано на следующем рисунке.



### Размещение статического текста


Теперь в рамке, размещенной под емкостью, сделаем две надписи: **Уровень** и **Давление**. Для этого нажмем ЛК на иконке статического текста в инструментальной панели элементов рисования.

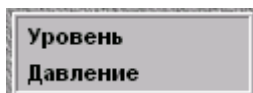


При этом на экран выводится диалог настройки атрибутов.



Нажмем ЛК на кнопке настройки параметров шрифта. Установим в появившемся диалоге шрифт **Arial**, начертание – **полужирный**, размер – **8**, а набор символов – **кириллица**. Зададим цвет символов черным и выравнивание по левому краю. Наберем в окне ввода текста слова **Уровень** и **Давление**. После этого переведем курсор в область экрана и разместим текст.

 Теперь перейдем в режим редактирования и переместим введенный текст, чтобы он был расположен так, как показано на рисунке.



## Отображение в графическом виде значений каналов

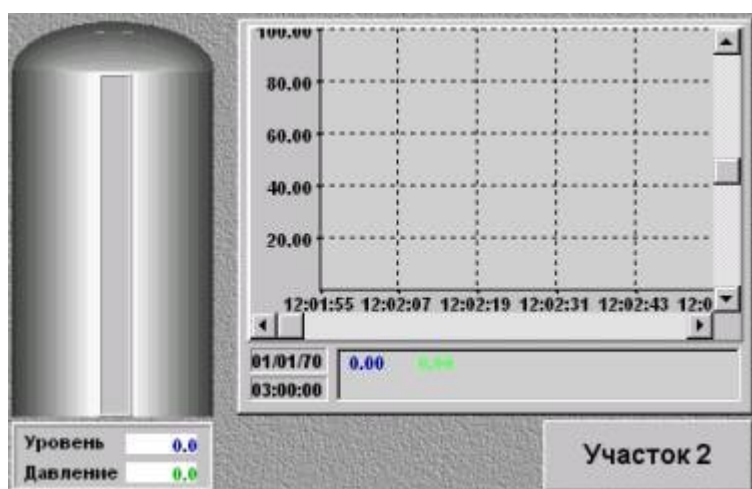
[Отображение уровня](#)

[Вывод числовых значений параметров](#)

[Тренды параметров](#)

[Переход на другой экран](#)

Перейдем теперь к размещению динамических элементов. Окончательный вид экрана **Участок 1** показан на следующем рисунке.

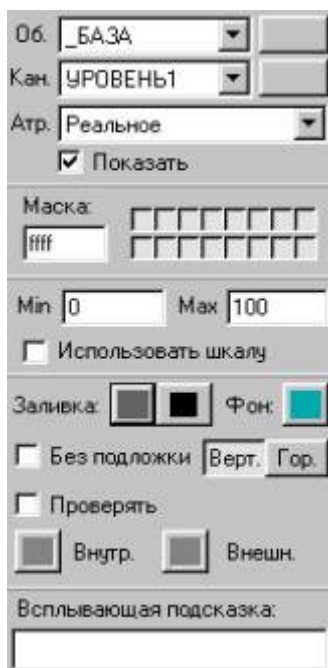


## Отображение уровня

Чтобы показать уровень заполнения емкости, используем форму отображения **Гистограмма**. Выберем ее нажатием ЛК на соответствующей иконке инструментальной панели.



При этом на экране появится диалог настройки атрибутов данной формы отображения. Его вид показан на следующем рисунке.



Настроим атрибуты гистограммы, как показано на рисунке. Далее свяжем ее с реальным значением канала **УРОВЕНЬ1** из объекта базы каналов **\_БАЗА**. После этого поместим гистограмму внутри рамки, расположенной по центру емкости.

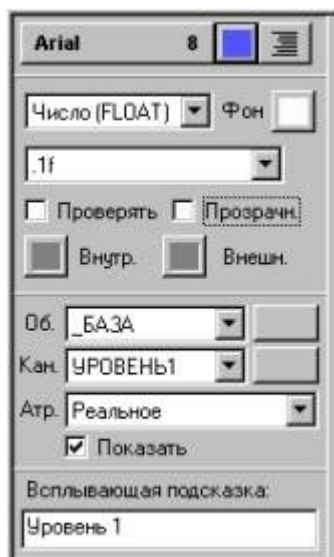
### Вывод числовых значений параметров

Для вывода значений параметров в виде чисел предназначена текстовая форма отображения. Для ее размещения нажмем ЛК на соответствующей иконке инструментальной панели форм отображения, показанной на следующем рисунке.



При этом на экране появится диалог настройки атрибутов данной формы отображения. Его вид показан на следующем рисунке.

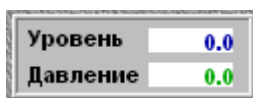
Настроим атрибуты текстовой формы, как показано на рисунке. При этом цвет символов установим темно-синим, а фона - светло-серым.



Свяжем данную форму отображения с реальным значением канала **УРОВЕНЬ1** из объекта **\_БАЗА** и разместим ее на экране под изображением емкости напротив слова **Уровень**.

Затем сменим цвет символов в диалоге настройки атрибутов на светло-зеленый, свяжем форму с каналом **ДАВЛЕНИЕ1** и разместим новую форму отображения напротив слова **Давление**.

Рамка, расположенная под изображением емкости, будет выглядеть следующим образом.



## Тренды параметров

Вывод трендов технологических параметров осуществляется с помощью специальной формы отображения.

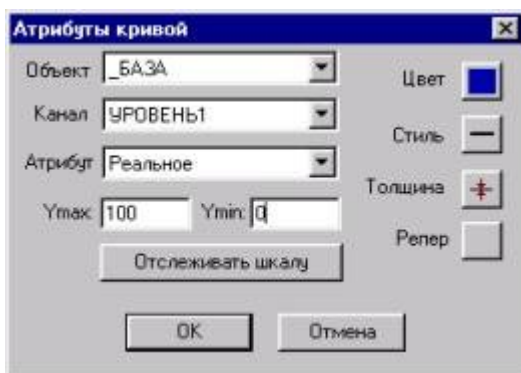


Для ее размещения нажмем ЛК на соответствующей иконке инструментальной панели форм отображения, показанной на рисунке.

При этом на экране появится диалог настройки атрибутов трендов. Вид этого диалога показан на следующем рисунке.



Настроим оси, сетку и шрифт для надписей по осям графика, как показано на рисунке и перейдем к настройке линий графика. Для этого нажмем ЛК на кнопке **Добав**. При этом на экран выводится диалог **Атрибуты кривой**, показанный на следующем рисунке.



Свяжем эту кривую с реальным значением канала **УРОВЕНЬ1**. Далее настроим диапазон вывода, цвет, стиль и толщину линии, как показано на рисунке. Подтвердим завершение настройки кривой нажатием ЛК на кнопке **OK**.

Повторим операции по настройке кривой. На этот раз свяжем ее с каналом **ДАВЛЕНИЕ1**, а цвет линии зададим светло-зеленый.

Разместим тренд так, как показано на рисунке в начале раздела.

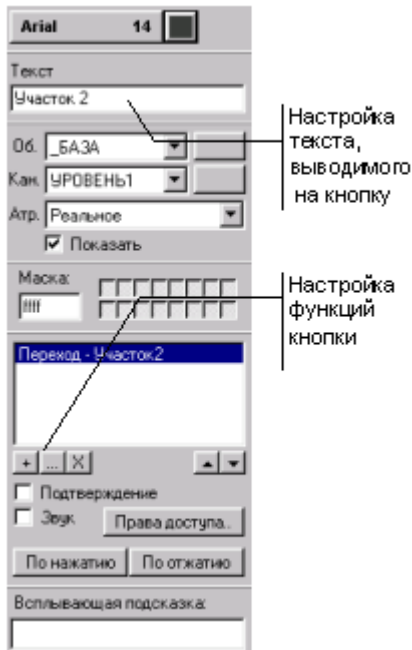
### Переход на другой экран

Последней формой отображения, которую мы разместим на экране, будет кнопка, реализующая переход на экран **Участок 2**.

Чтобы создать и настроить эту форму отображения, нажмем ЛК на соответствующей иконке инструментальной панели форм отображения, показанной на следующем рисунке.

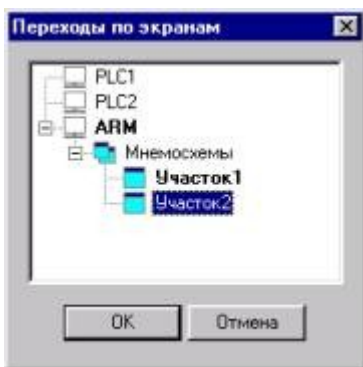


- Эта группа включает в себя четыре формы отображения. После повторного нажатия ЛК на той же иконке в инструментальной панели выберем в появившемся меню **кнопку с текстом "мягкую"**. Окно **Атрибуты** для этой ФО показано на следующем рисунке.



Настроим текст для вывода на кнопку, как показано на рисунке.

Далее настроим функцию перехода на другой экран. Для этого нажмем ЛК на кнопке [+] под окном функций и выберем из меню функцию **Переход**. На экране появляется следующий диалог.



Выберем экран **Участок 2** и нажмем ЛК на кнопке ОК. После этого установим флаг выполнения функции перехода. Это осуществляется нажатием ЛК в области, расположенной в начале строки настройки переходов. После этого разместим кнопку на экране.

На этом разработка экрана **Участок 1** закончена.

### Тиражирование графики



## Выделение копируемых элементов

## Копирование фрагмента экрана в буфер обмена


## Вставка графических элементов из буфера обмена

## Модификация форм отображения

Второй технологический участок идентичен первому. Информация на экране **Участок 2** будет представляться в том же виде, что и на экране **Участок 1**. Поэтому скопируем рисунок с первого экрана на второй и затем свяжем формы отображения с другими каналами.


Для копирования рисунка надо выделить все копируемые графические элементы, затем поместить их в буфер обмена. После этого следует перейти на нужный экран и вставить содержимое буфера.

### **Выделение копируемых элементов**

 Для выделения рисунка на экране **Участок 1** перейдем в режим редактирования. Затем нажмем ЛК в левом верхнем углу экрана и, удерживая кнопку мыши в нажатом состоянии и перемещая курсор, обведем контурным прямоугольником весь рисунок. После этого кнопку мыши можно отпустить. Контурный прямоугольник примет минимальный размер для охвата всех полностью попавших в него элементов. Эту же операцию можно выполнить командой **Выделить все** из меню **Правка**.


### **Копирование фрагмента экрана в буфер обмена**

Для копирования выделенных элементов в буфер обмена следует выполнить одно из следующих действий:

- выполнить команду **Копировать** из меню **Правка**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **CTRL-C**.

### **Вставка графических элементов из буфера обмена**

После копирования рисунка в буфер обмена надо перейти на экран **Участок 2** и выполнить одну из следующих операций:

- выполнить команду **Вставить** из меню **Правка**;
- нажать ЛК на иконке  инструментальной панели;
- нажать сочетание клавиш **CTRL-V**.

После вставки графических элементов из буфера они объединены общим выделением. Следует переместить вставленный рисунок, чтобы он был расположен так же, как на экране **Участок 1**. Чтобы снять с рисунка групповое выделение, надо перейти в режим

размещения графических элементов. Это делается выбором любого элемента рисования или формы отображения в соответствующей инструментальной панели.

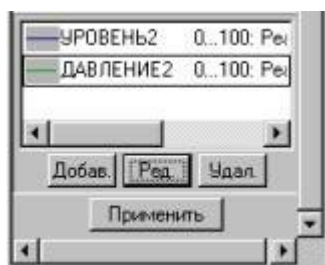
## Модификация форм отображения

Модифицируем теперь формы отображения на экране **Участок 2**.

Для изменения настроек форм отображения следует перейти в режим редактирования. Для этого надо нажать ЛК на соответствующей иконке системной инструментальной панели.

Далее при выделении любого графического элемента на экране появляется диалог настройки его атрибутов. В него копируются значения атрибутов выделенного элемента.

В отличие от режима размещения новых элементов, в нижней части этого диалога имеется кнопка **Применить**. При нажатии ЛК на ней все внесенные изменения атрибутов графического элемента, будут зафиксированы.



Выделим гистограмму, отображающую уровень в емкости. Нам требуется привязать ее к каналу **Уровень2**. Так же привяжем остальные формы отображения связанные с каналов **Уровень1** на канал **Уровень2**, а связанные с каналом **Давление1** объекта – на канал **Давление2**. Для изменения настроек графиков следует выбрать нужную кривую и нажать ЛК на кнопке **Ред.**

Кроме того, надо сменить экран для перехода и надпись на кнопке. Эта кнопка теперь должна управлять переходом на экран **Участок 1**. Надпись на этой кнопке должна воспроизводить имя экрана, на который будет осуществляться переход.

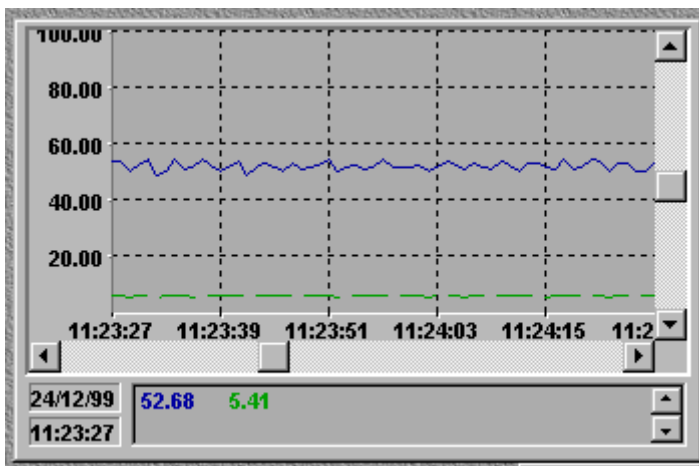
## Эмуляция работы графической базы

На этом разработка графического интерфейса для данной задачи закончена. Для проверки функционирования размещенных на экранах форм отображения следует перейти в режим эмуляции.

Переход в режим эмуляции реализуется нажатием ЛК на иконке системной инструментальной панели, показанной на следующем рисунке.



Повторное нажатие этой кнопки позволяет остановить пересчет базы каналов. В окне тренда будут выведены кривые графиков давления и уровня.



## ЛР 5

Эта ЛР посвящена организации архивирования данных. В ТРЕЙС МОУД для этого предусмотрены три типа архивов, а также поддержка связи с базами данных через ODBC. В рамках ЛР мы создадим в проекте БЫСТРЫЙ\_СТАРТ2 локальный архив СПАД и отчет тревог. При этом будут рассмотрены следующие темы:

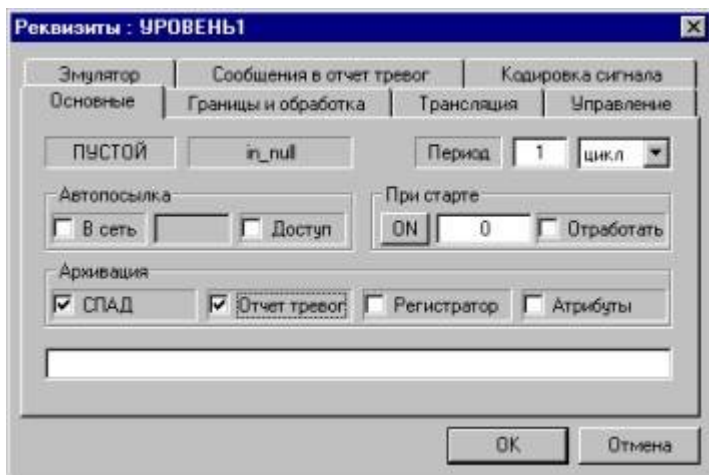
- Настройка каналов для архивирования;
- Настройка параметров СПАД;
- Настройка параметров отчета тревог;
- Визуализация архивных данных.

### Настройка каналов для архивирования

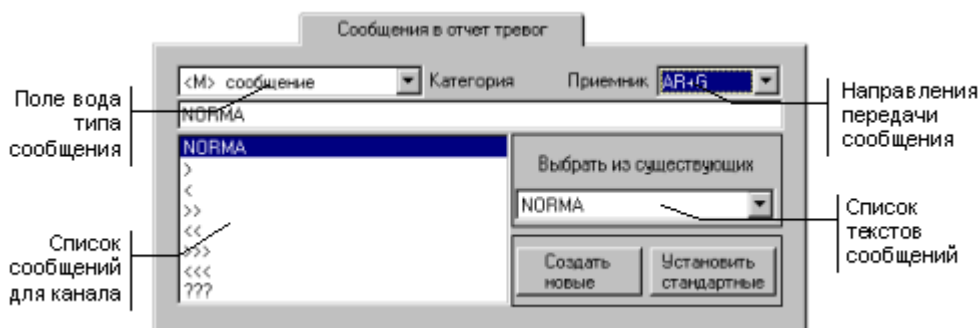
Загрузим проект БЫСТРЫЙ\_СТАРТ2 в редактор базы каналов. Настроим сохранение значений каналов узла **ARM** в отчет тревог и локальный архив, реализующий технологию СПАД. Для этого выделим данный узел в окне структуры проекта и войдем в окно редактирования его базы каналов.

Чтобы значения каналов записывались в архивы, им надо установить соответствующие флаги. Войдем в диалог **Каналы объекта** для объекта **\_БАЗА**. Для этого дважды нажмем ЛК на его изображении.

Войдем в диалог **Реквизиты** для канала **УРОВЕНЬ1**. Для этого дважды нажмем ЛК на его имени в списке диалога Каналы объекта. В бланке **Основные** диалога **Реквизиты** следует установить флаги **СПАД** и **Отчет тревог** раздела **Архивация**, как это показано на рисунке.



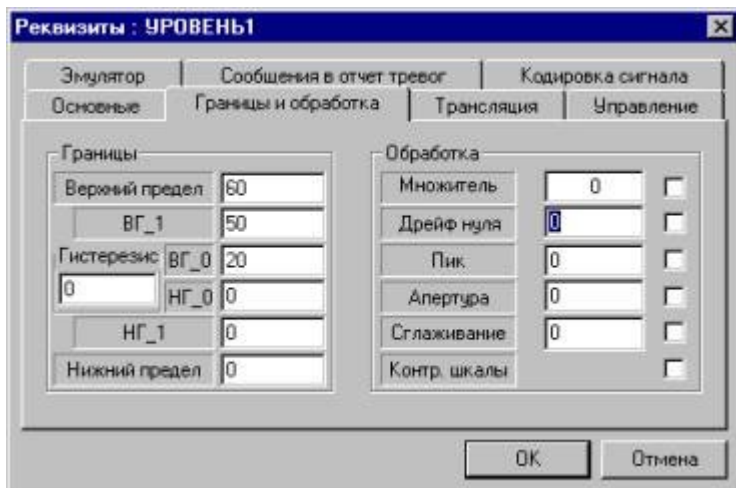
Тексты сообщений для отчета тревог задаются в бланке **Сообщения в отчет тревог** диалога **Реквизиты**.



Для каналов, контролирующих аналоговые параметры можно настроить восемь сообщений. Среди них: сообщение о регламентном состоянии, шесть сообщений о нарушении границ и сообщение о недостоверности данных. Для каналов, контролирующих дискретные параметры можно задать по два сообщения на каждый сигнал.

Для каждого канала можно задать собственные сообщения или использовать стандартные. Воспользуемся последними.

Настроим границы канала **УРОВЕНЬ1** следующим образом.



Эти же операции, за исключением настройки сохранения в отчет тревог, нужно проделать и с другими каналами данного объекта.

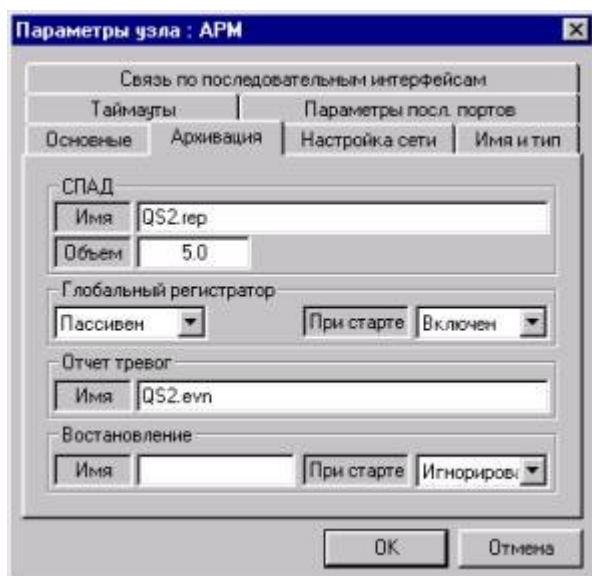
### Настройка параметров архивов

Помимо установки флагов для каналов необходимо указать параметры соответствующих архивов, которые будет вести узел ARM. Эти параметры настраиваются в бланке **Архивация** диалога **Параметры узла**. Чтобы войти в него, перейдем в окно структуры проекта и нажмем ПК на графическом изображении узла **ARM**.

Параметры локального архива настраиваются в разделе СПАД. Здесь в поле **Имя** следует ввести имя файла локального архива, а в поле **Объем** – размер этого файла в мегабайтах. Зададим имя **QS2.rep**, а размер равным 5 Мбайтам. При превышении этого объема происходит циклическая перезапись данных.

Для отчета тревог также нужно указать в соответствующем поле имя файла и, если это требуется, путь к нему.

Настройки архивов демонстрируются на следующем рисунке.



На этом настройка архивирования в редакторе базы каналов закончена. Сохраним проект и выйдем из редактора. Далее в графической базе узла ARM надо создать еще один экран, на котором следует разместить специальные формы просмотра архивных данных.

### Просмотр архивных данных

Загрузим проект **БЫСТРЫЙ\_СТАРТ2** в редактор представления данных и откроем графическую базу узла **ARM**. Для этого дважды нажмем ЛК на его имени в навигаторе проекта. Выделим экран **Участок 2** и нажмем ПК на его имени. В появившемся меню выполним команду **Добавить экран**. Присвоим новому экрану имя **История**.

Откроем новый экран для редактирования. Чтобы просматривать данные, сохраненные в архивах, разместим на нем архивный тренд и форму отображения для просмотра сообщений в отчете тревог.

В конечном виде при включенной эмуляции этот экран должен выглядеть следующим образом.



### Просмотр локального архива

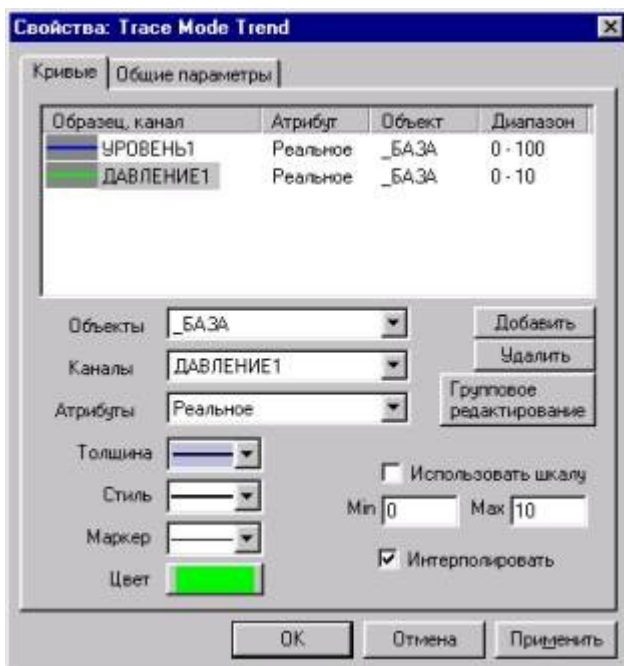
Для просмотра данных из локального архива воспользуемся формой отображения **Универсальный тренд**. Нажмем дважды ЛК на иконке трендов панели форм отображения.



Выберем в появившемся меню универсальный тренд.

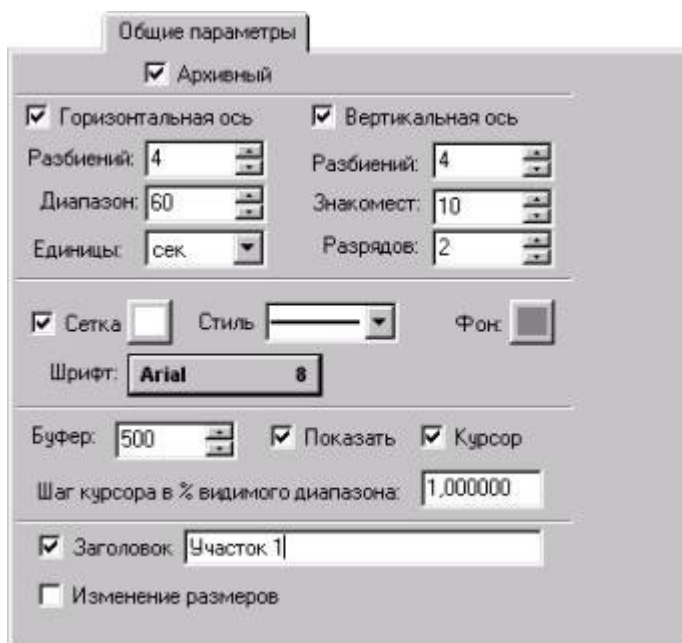
 - универсальный тренд.

Выберем универсальный тренд и разместим его на экране. Появится диалог настройки его атрибутов, показанный на следующем рисунке.



Чтобы добавить в список отображаемых параметров новую кривую, нажмем ЛК на кнопке **Добавить**. После этого свяжем кривую с реальным значением канала **УРОВЕНЬ1**, цвет линии установим синий, а диапазон вывода от 0 до 110. Затем нажмем еще раз ЛК на кнопке **Добавить** и свяжем новую кривую с каналом **ДАВЛЕНИЕ1**. Поменяем цвет на светло-зеленый, а диапазон зададим от 0 до 10.

Далее откроем бланк **Общие параметры** того же диалога и установим параметры тренда, как показано на следующем рисунке.

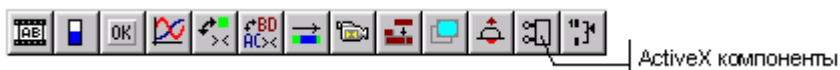


Завершим настройку нажатием ЛК на кнопке **ОК**.

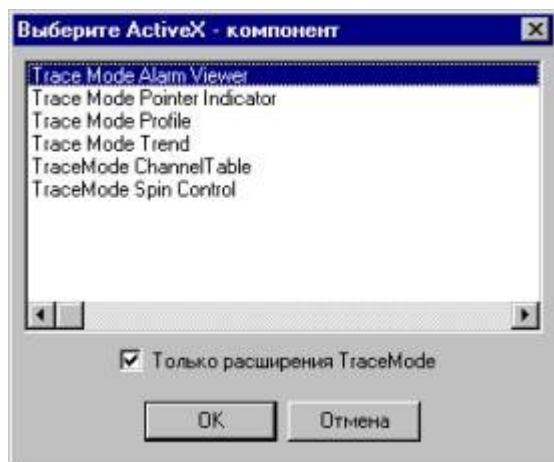
### Просмотр отчета тревог

- Для размещения формы просмотра отчета тревог нужно на панели форм отображения нажать кнопку **ActiveX компоненты**.





При этом появится следующий диалог.



Установим в нем флаг **Только расширения Trace Mode**, выберем из списка компонент **Trace Mode Alarm Viewer** и разместим его в верхней половине экрана.

После этого запустим режим эмуляции нажатием ЛК на соответствующей кнопке системной инструментальной панели.



В момент запуска архивный тренд данных отображать не будет, так как архив пуст. Подождите секунд 15, нажмите ЛК в области тренда, а затем клавишу **Home**. Тренд отобразит данные из архива. После того, как по количеству данных в архиве превысит диапазон отображения тренда, станет доступна нижняя линейка прокрутки.

Для просмотра сообщений, заносимых в отчет тревог, в реальном времени, нажмите ПК в разделе **События** формы **Alarm Viewer** и установите в появившемся на экране меню флаг **В реальном времени**.

На этом ЛР 5 закончена.

## ЛР 6

Эта ЛР посвящена организации документирования технологических параметров проекта. В ТРЕЙС МОУД для решения данной задачи используется специальный модуль – **сервер документирования**. Этот модуль в состав стандартной поставки инструментальной системы ТРЕЙС МОУД не входит. По команде от мониторов реального времени, собственному сценарию или по команде от оператора сервер документирования интерпретирует созданные заранее шаблоны, запрашивает у МРВ необходимые данные и формирует по ним готовые документы. Для создания шаблонов документов в инструментальную систему включен специальный редактор – **редактор шаблонов**.

Шаблон документа разрабатывается в виде файла HTML-формата. В него могут быть вставлены любые элементы, поддерживаемые в HTML, а также дополнительные функции



и команды, предназначенные для запроса данных от узлов проекта ТРЕЙС МОУД и обработки полученных значений.

В рамках этой ЛР в проекте БЫСТРЫЙ\_СТАРТ2 мы будем каждые 5 минут формировать документ, содержащий информацию об уровне жидкости в емкостях. Текущие значения уровня контролируются каналами **УРОВЕНЬ1** и **УРОВЕНЬ2** объекта **\_БАЗА** узла **ARM**.

### *Разработка шаблона.*



Запустим редактор шаблонов, дважды нажав ЛК на соответствующем ярлыке в программной группе ТРЕЙС МОУД. Другой способ – запустить файл **htmpled.exe**, который находится в директории инструментальной системы.

Создаваемый шаблон необходимо привязать к проекту, из которого сервер документирования будет брать данные для итогового документа. Для этого из меню **Файл** выполним команду **Выбрать проект** и укажем в появившемся диалоге файл **БЫСТРЫЙ\_СТАРТ2.ctm**. Далее сохраним шаблон, выполнив команду **Сохранить** из этого же меню. В появившемся на экране диалоге укажем папку инсталляции сервера документирования, а имя файла зададим **Уровень.html**.

Далее поместим курсор в первую строку шаблона, затем нажмем ЛК на иконке выбора стиля и выберем стиль **Заголовок 1**.



– выбор одного из стандартных стилей текста.

После этого наберем текст **Уровень жидкости в емкостях**. Нажмем клавишу **<Enter>** и перейдем на следующую строку. Установим для этой строки стиль **Заголовок 2**.

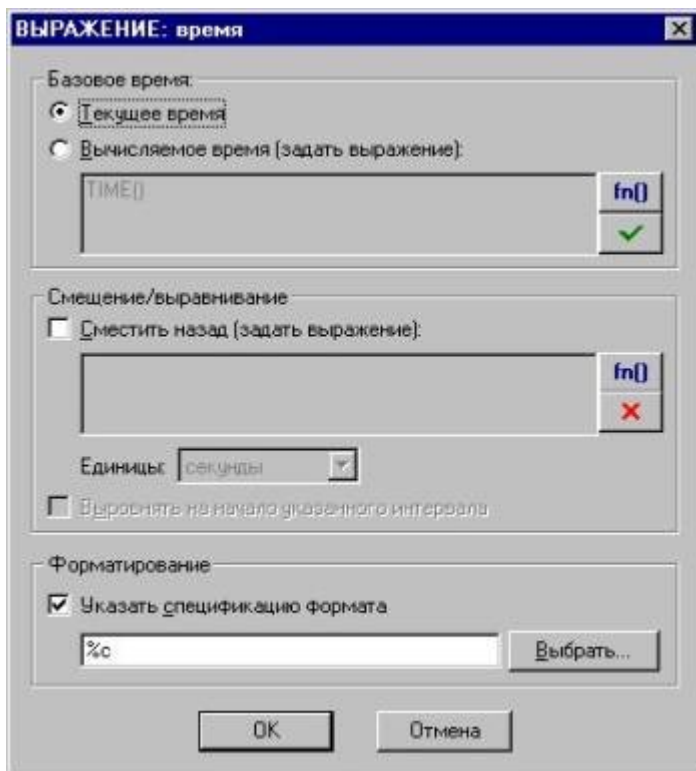
### **Дата подготовки документа**

Во второй строке требуется вывести дату и время формирования документа. Для этого нажмем ЛК на иконке инструментальной панели элементов, предназначенной для вставки времени.

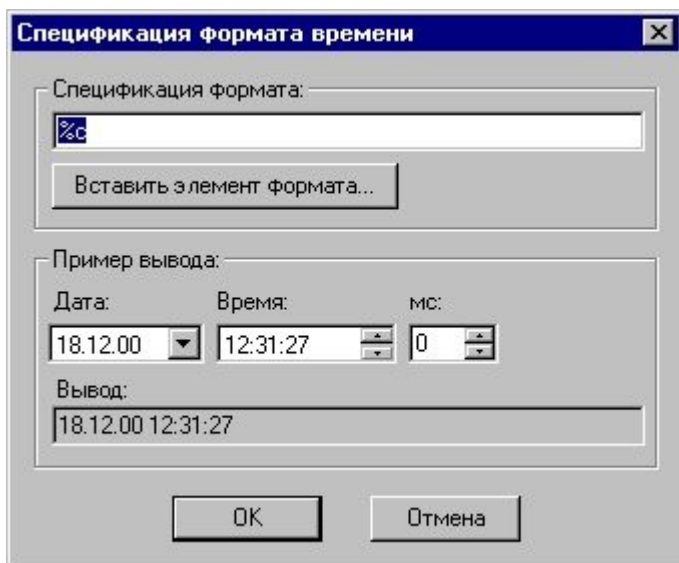


– вставить поле вывода даты или времени.

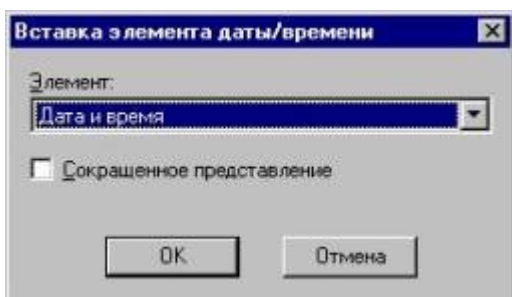
При этом на экране появится диалог **ВЫРАЖЕНИЕ: время:**



Укажем текущее время в качестве базового в соответствующем разделе диалога. Затем для задания формата вывода даты и времени следует нажать ЛК на кнопке **Выбрать** раздела **Форматирование**. На экране появится следующий диалог:



В этом диалоге надо стереть содержимое строки **Спецификация формата**, нажать ЛК на кнопке **Вставить элемент формата** и в появившемся диалоге выбрать элемент **Дата и время**:



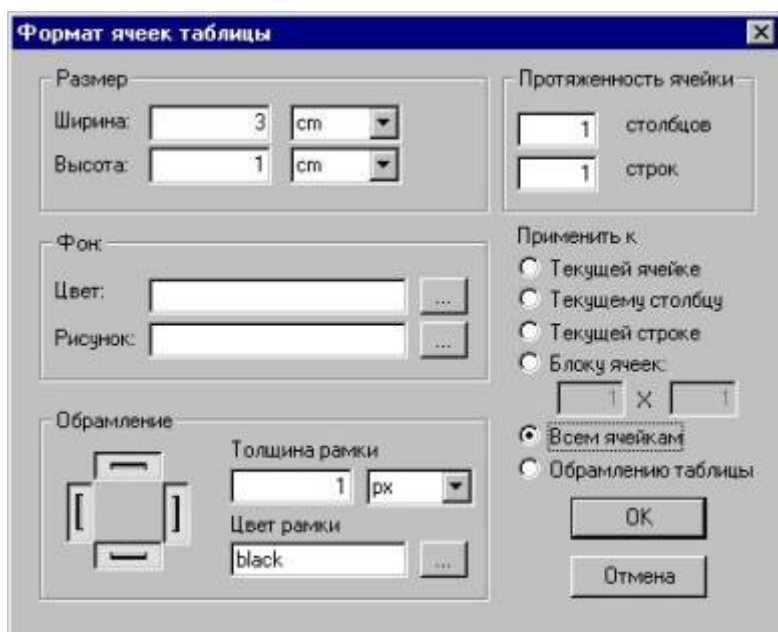
После нажатия ОК во всех открытых диалогах на второй строке шаблона появится значок **f(x)**, а вывод даты и времени в сгенерированном документе будет выглядеть следующим образом: **19 декабря 2000 г 12:34:12**.

### Таблица с данными

После вставки во второй строке выражения для вывода даты и времени перейдем на следующую строку нажатием клавиши **<Enter>**. Стиль для этой строки установим **Абзац**. Далее вставим в нее таблицу. Для этого нажмем ЛК на специальной иконке в инструментальной панели.

- – вставить таблицу.

Установим курсор в любую из ячеек таблицы, нажмем ПК и выполним команду **Формат ячейки** из появившегося меню. На экране появится диалог **Формат ячеек таблицы**, в котором нужно произвести настройку параметров, как показано на следующем рисунке:




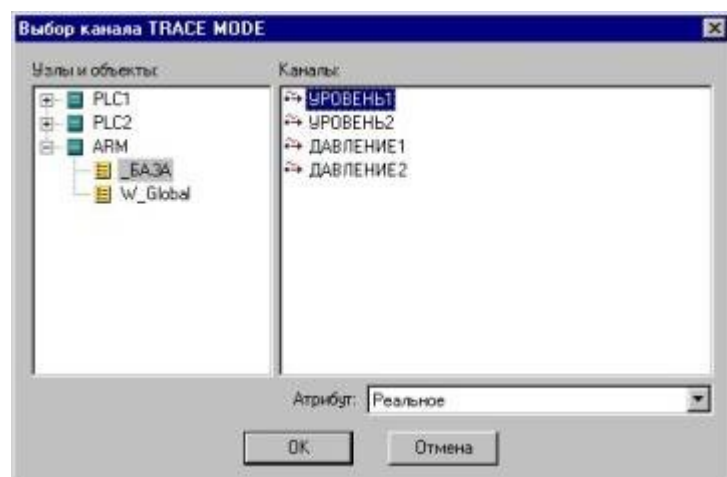
Далее в первой ячейке левого столбца таблицы разместим текст **Уровень1, %**, во второй – **Уровень2, %**.

В ячейках правого столбца при генерации документа должны выводиться значения каналов **Уровень1** и **Уровень2**. Для этого в шаблоне в эти ячейки нужно вставить специальные выражения.

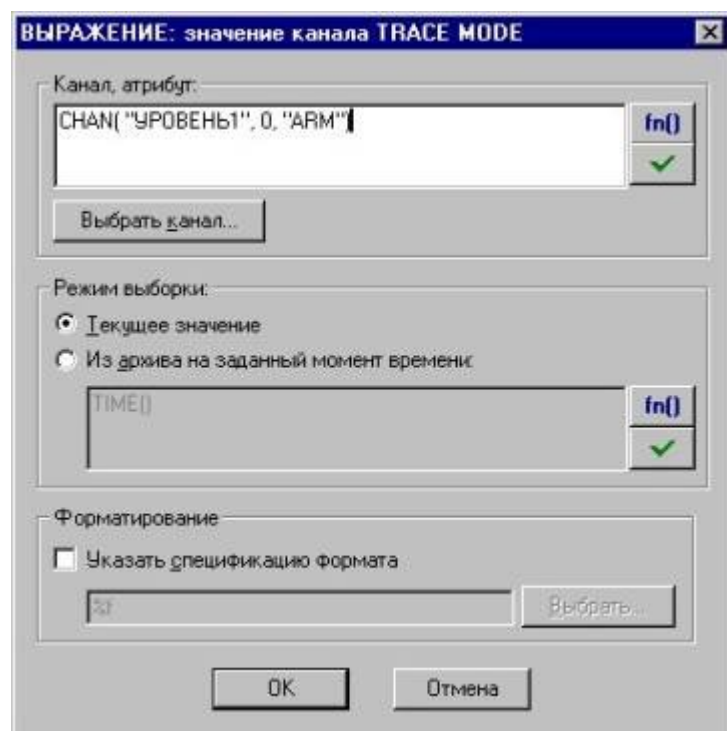
Установим курсор в верхнюю правую ячейку таблицы и выполним команду вставки выражения в шаблон:

 – вставить произвольное выражение.

В появившемся на экране диалоге нажмем кнопку  и выполним команду **Значение канала** из контекстного меню. На экране появится следующий диалог:

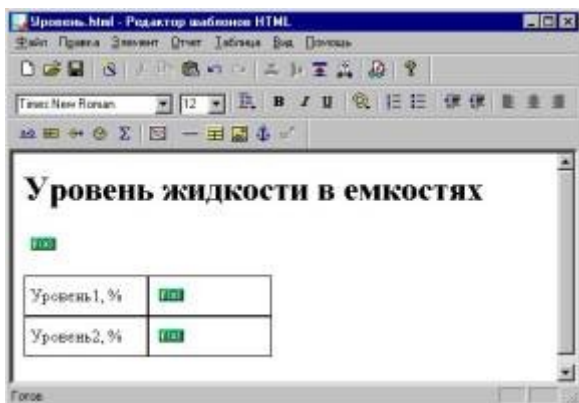


В этом диалоге нужно выбрать канал **Уровень1** объекта **\_БАЗА** узла **ARM**, указать атрибут **Реальное** и нажать **ОК**. На экране появится диалог, в разделе **Канал, атрибут** которого в нотации языка **Техно LIST** редактора шаблонов отображается выражение для вывода выбранного значения канала в ячейку таблицы. В разделе **Режим выборки** этого диалога установим флаг **Текущее значение**:

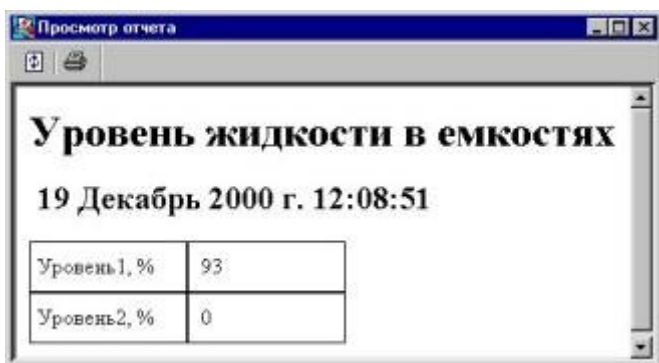


Нажмем **ОК** во всех открытых диалогах и выполним аналогичную процедуру для второй ячейки правого столбца, выбрав для вывода в нее реальное значение канала **Уровень2**.

После всех настроек шаблон должен выглядеть следующим образом:



Сохраним его и выполним команду **Пробный отчет** из меню **Отчет** редактора. Данная команда позволяет проверить правильность создания шаблона путем эмуляции пробного документа, при этом вместо реальных данных, полученных от МРВ, в таблицу заносятся значения, сгенерированные редактором шаблонов:



### *Создание сценария и генерация документа*

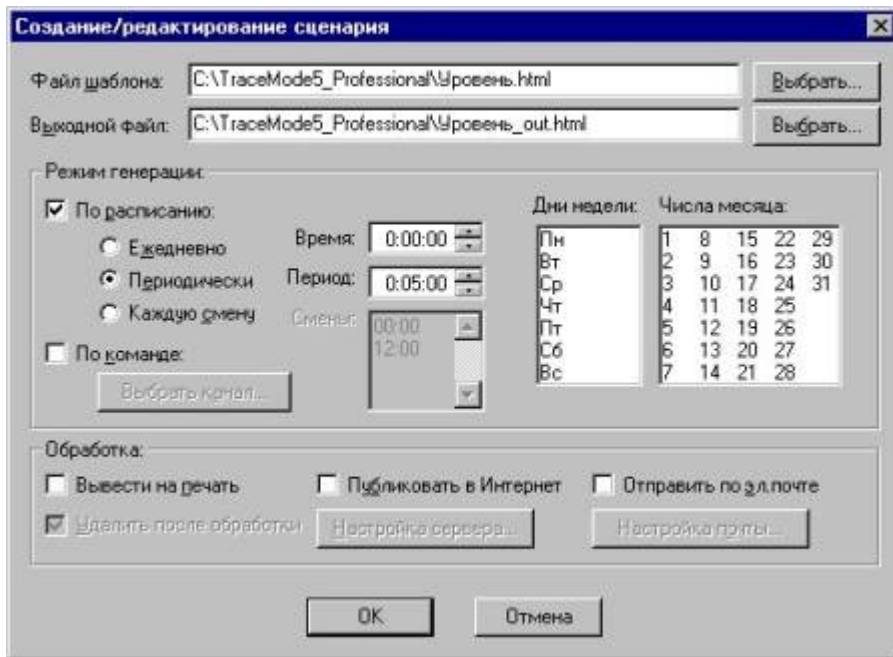
Для создания сценария генерации документа по созданному шаблону запустим **сервер документирования**, нажав ЛК на соответствующем ярлыке в программной группе ТРЕЙС МОУД. Другой способ – запустить файл **docserv.exe**, который находится в директории установки сервера.

На экране появится окно сервера документирования. На инструментальной панели нажмем кнопку создания нового сценария.

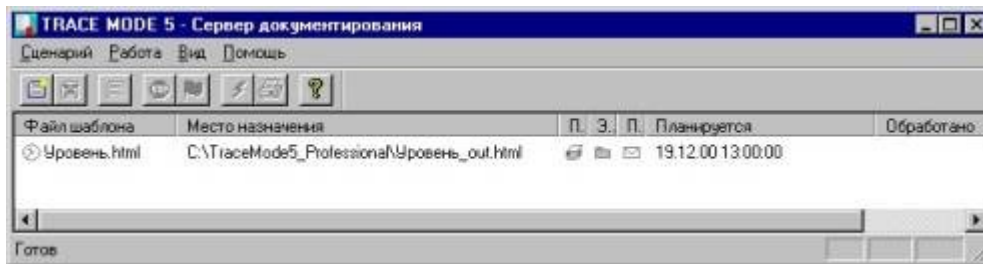


– создать новый сценарий.

В появившемся диалоге в качестве файла шаблона следует указать **Уровень.html**. Далее в разделе **Режим генерации** надо установить флаг **По расписанию**, выбрать пункт **Периодически** и задать для периода значение 5 минут. Значение поля **Время** надо оставить нулевым. После указанных настроек документ будет создаваться каждые 5 минут и сохраняться в файле **Уровень\_out.html**:



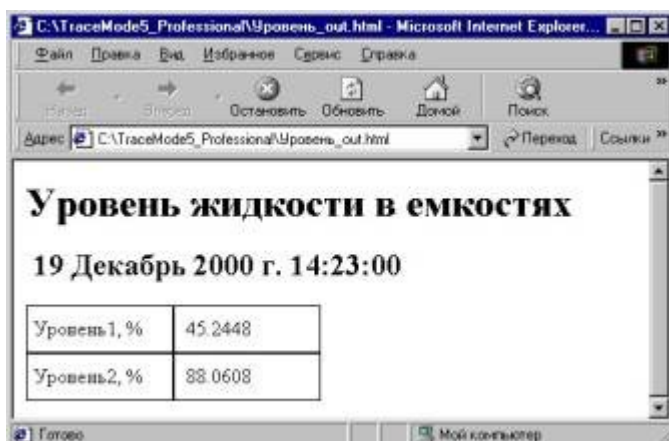
После нажатия кнопки ОК в окне сервера документирования появится строка с описанием созданного сценария:



Чтобы сервер документирования получал реальные данные для генерируемого документа, запустим редактор представления данных, загрузим в него проект БЫСТРЫЙ\_СТАРТ2 и

перейдем в режим эмуляции, нажав ЛК на кнопке .

По истечении времени, указанного в разделе **Планируется** окна сервера документирования, файл **Уровень\_out.html** будет создан и в дальнейшем будет перезаписываться каждые 5 минут. Примерный вид генерируемого документа в редакторе HTML показан на следующем рисунке:



## ЛР 7

Эта ЛР посвящена организации управления технологическим процессом через Интернет с помощью модуля **Web-активатор** исполнительной системы ТРЕЙС МОУД.

В рамках ЛР рассматривается управление проектом **Шихтоподготовка** (этот проект входит в состав инструментальной системы) при запуске Web-браузера и сервера матобработки ТРЕЙС МОУД на локальном компьютере, однако все описанные ниже процедуры справедливы и для реализации режима управления технологическим процессом с удаленного компьютера.

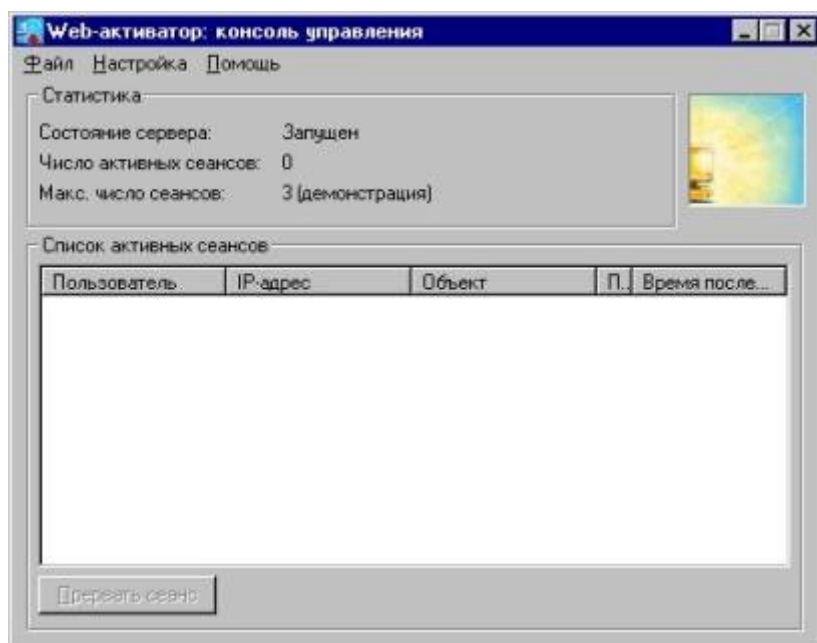
Для решения задачи нам потребуется сконфигурировать сервер HTTP и создать сайт проекта.

### Конфигурация сервера HTTP

Для установки и конфигурации сервера HTTP предназначена консоль управления Web-активатора. Для ее запуска нужно нажать ЛК на соответствующем ярлыке в программной группе ТРЕЙС МОУД или запустить файл **Web\_Activator.exe**, находящийся в директории инсталляции Web-активатора.

На экране появится окно консоли, при этом сервер будет запущен:

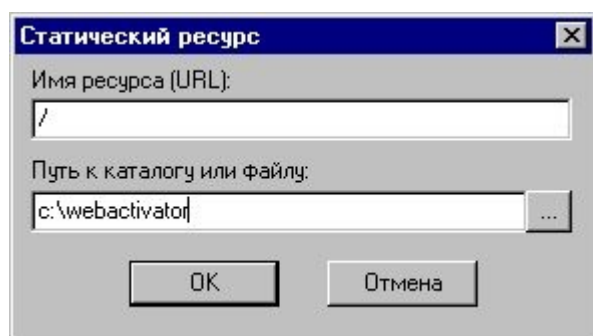
Консоль отображает текущее состояние сервера. Если сервер не запущен, выполните последовательно команды **Установить сервер** и **Запустить сервер** из меню **Файл/Управление сервером**.



Из меню **Настройка** консоли управления выполним команду **Статические ресурсы** и в появившемся на экране диалоге **Настройка статических ресурсов** нажмем кнопку **Создать**. На экране появится диалог, в котором нужно задать имя директории, в которой в дальнейшем будут размещены ресурсы сайта, а также в окне **Имя ресурса (URL)** присвоить этой директории адрес (URL). Имя ресурса должно начинаться со знака / . Если указанной директории не существует, она будет создана. Кроме того, в этой директории

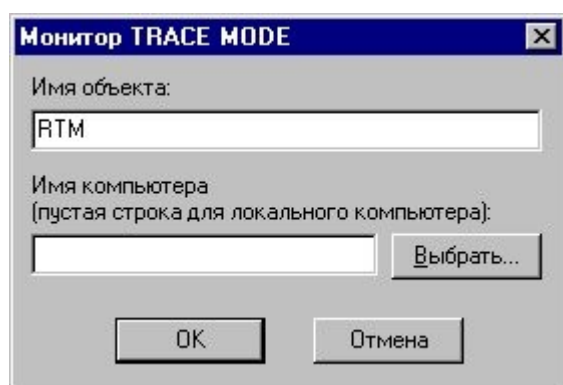


будет создан файл **index.htm** (файл с таким именем является по умолчанию первой страницей сайта). Выполним настройки, как показано на рисунке, и нажмем ОК:



В появившемся на экране диалоге с приглашением отредактировать файл **index.htm** нажмем кнопку **Нет**, в диалоге **Настройка статических ресурсов** – кнопку **Выход**.

Далее следует задать имя объекта, у которого сервер будет запрашивать данные. Для этого выполним команду **Мониторы TRACE MODE** из меню **Настройка** консоли управления Web-активатора и в диалоге **Мониторы TRACE MODE** нажмем кнопку **Создать**. В появившемся на экране диалоге введем имя объекта, как показано на рисунке, и нажмем ОК:



В диалоге **Мониторы TRACE MODE** нажмем кнопку **Выход**.

После настроек сервера HTTP перейдем к созданию сайта проекта.

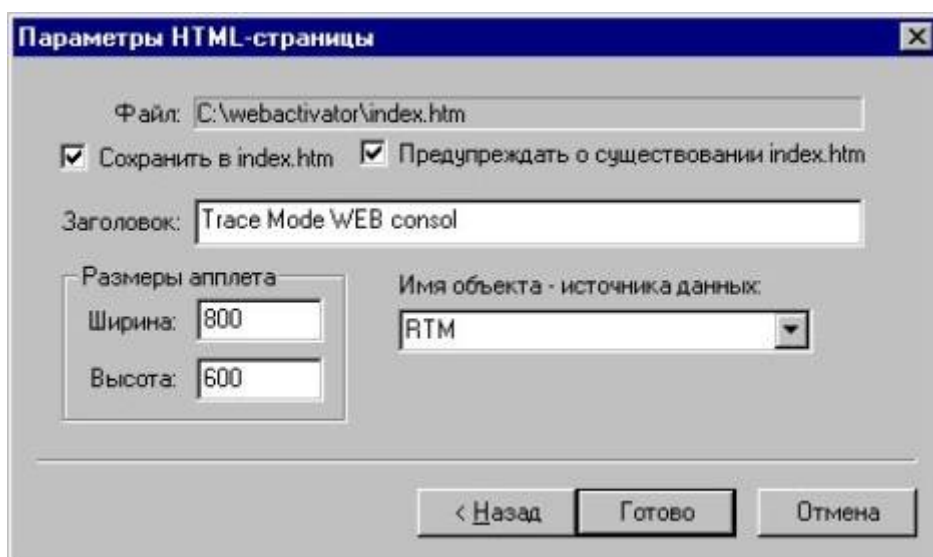
### **Создание сайта проекта на сервере**

В созданную директорию ресурсов нужно экспортировать из редактора представления данных графическую базу узла в формате **xml** и создать в этой директории HTML-файл, содержащий теги для отображения графической базы в Web-браузере. Для этого нужно загрузить проект в редактор (файл **charge.ctm** находится в поддиректории **Samples** директории установки инструментальной системы), в бланке **Экраны** навигатора проекта нажать ПК на имени узла (**АРМ\_оператора**), установить курсор на команде **Экспорт** и из открывшегося контекстного меню выполнить команду **Для Web-активатора**. На экране появится диалог сохранения файла в формате **xml**. В этом диалоге зададим для файла имя **charge.xml**, выберем для его размещения созданную директорию ресурсов **C:\webactivator** и нажмем кнопку **Сохранить**.

В последующих диалогах установки параметров экспорта графической базы узла **Качество сохранения изображений**, **Архивирование JPG файлов**, **Размещение JPG**




файлов сохраним настройки, заданные по умолчанию, а в диалоге. **Параметры HTML страницы** произведем настройки, как показано на рисунке:



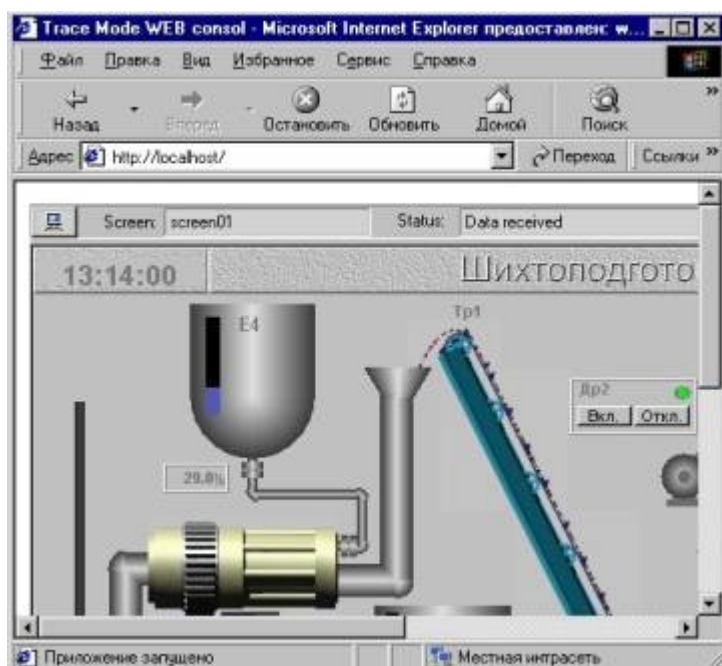
Нажмем кнопку **Готово** и в появившемся на экране диалоге-предупреждении о существовании в указанной директории файла **index.htm** подтвердим перезапись этого файла.

После этого в директории **C:\webactivator** будет размещена графическая база узла, а файл **index.htm** будет содержать теги для отображения графической базы в Web-браузере.

Далее перейдем в режим эмуляции, нажав кнопку  на инструментальной панели редактора представления данных.

### Управление техпроцессом через Интернет


Запустим Web-браузер и зададим для соединения адрес созданного сайта: **http://localhost/**. При этом в обозревателе отображается графическая база узла:




Для реализации данного режима рекомендуется использовать JVM либо от MS (IE 5,6), либо от Sun (Java 2 версия 1.4 или старше).

Для реализации данного режима необходимо произвести настройку параметров DCOM (см. [Настройка DCOM](#) раздела **Работа в реальном времени**).

При управлении проектом с удаленного компьютера вместо **localhost** должен быть указан IP-адрес компьютера, на котором запущен **Web-активатор**.

Для подключения к проекту нажмем ЛК на иконке . В появившемся диалоговом окне введем **default** в полях имени пользователя и пароля и нажмем кнопку **ОК**. После этого управление техпроцессом из Web-браузера становится доступным.

Для окончания сеанса надо нажать ЛК на иконке  и закрыть окно Web-браузера.

## Лабораторные работы в Labview 6

### 1. Возможности и характеристики современных SCADA-систем

### 2. Основы разработки ППО в среде программирования LabVIEW

#### 2.1. Основные понятия и панели LabVIEW

#### 2.2. Использование структур и построение графиков

#### 2.3. Массивы и кластеры

#### 2.4. Строки, таблицы, файлы

#### 2.5. Подпрограммы

### 3. Разработка VI для функционирования в сети Internet

### 4. Взаимодействие vi по протоколу TCP

### 5. Использование изображений

### 6. Сбор данных с реального объекта

### 7. Упражнения и лабораторные работы

### 8. Литература

#### 1. Возможности и характеристики современных SCADA-систем

Разработка и выбор специализированного прикладного программного обеспечения для создания автоматизированных систем управления определенным технологическим процессом (АСУ ТП), в том числе и для решения образовательных задач, осуществляется по двум возможным направлениям:

- разработка программ на основе базовых традиционных языков программирования;
- использование коммерческих инструментальных проблемно ориентированных средств.

Использование уникального программного обеспечения для каждого конкретного проекта хотя и может быть наиболее оптимальным с точки зрения решения определенной задачи, но необходимость каждый разрешать задачу, практически, с нуля, рост временных и материальных затрат существенно снижает их достоинства. В данной связи все большее предпочтение промышленными, коммерческими и образовательными

организациями отдается разработчикам специализированных операционных систем (ОС), аппаратного и программного обеспечения, предназначенных для конечных систем управления различными объектами типа SCADA-систем (от Supervisory Control And Data Acquisition – диспетчерское управление и сбор данных).

Перечислим некоторые популярные на западном и российском рынках SCADA-систем, имеющие поддержку в России [1]:

- Factory Link (United States DATA Co., USA );
- InTouch (Wonderware, USA);
- Genesis (Iconics, USA);
- WinCC (Siemens, Germany);
- Trace Mode (Ad Astra, Россия);
- RSView (Rockwell Software Inc, USA);
- LabVIEW, BridgeVIEW, LabVIEW RT, Lookout (National Instruments, USA) и др.

SCADA-системы, прежде всего, предназначены для получения и визуализацией информации от программируемых логических контроллеров (ПЛК), плат ввода-вывода информации, распределенных систем управления. Разработка на их основе комплексных, хорошо интегрированных инструментальных средств, обеспечивающих взаимодействие лабораторного оборудования различной степени сложности в автоматизированном режиме, позволяет реализовать на практике основные концепции использования современных информационно-коммуникационных технологий в образовательном процессе.

Рассмотрим основные возможности и характеристики современных SCADA-систем [1].

#### **Функциональные возможности.**

1. Разработка архитектуры всей системы автоматизации (на этом этапе определяется функциональное назначение каждого узла системы автоматизации).
2. Решение вопросов, связанных с возможной поддержкой распределенной архитектуры, необходимостью введения узлов с горячим резервированием и т.п.
3. Создание прикладной системы управления для каждого узла, где специалист в области автоматизируемых процессов наполняет узлы архитектуры алгоритмами, совокупность которых позволяет решать задачи автоматизации.
4. Приведение параметров прикладной системы в соответствие с информацией, которой обмениваются устройства нижнего уровня (ПЛС, АЦП, ЦАП) с внешним миром (датчиками температуры, давления и др.).

5. Отладка созданной прикладной программы в режиме эмуляции и реальном режиме.

### **Технические характеристики**

1. Программно-аппаратные платформы. Анализ перечня таких платформ необходим, поскольку от него зависит распространение SCADA-системы на имеющиеся вычислительные средства, а также оценивание стоимости ее эксплуатации. Подавляющее большинство SCADA-систем реализовано на MS Windows-платформах (Windows NT).
2. Имеющиеся средства сетевой поддержки. Для эффективного функционирования системы автоматизации распределенных объектов SCADA-система должна обеспечивать высокий уровень сетевого сервиса. Необходима поддержка сетевых сред с использованием стандартных протоколов (Netbios, TCP/IP и др.), а также наиболее популярных сетевых стандартов из класса промышленных интерфейсов (Profibus, Canbus, LON, Modbus и т.д.).
3. Встроенные командные языки. Большинство SCADA-систем имеют встроенные языки высокого уровня, Basic-подобные языки, для создания фрагментов алгоритма, необходимых в решении задачи управления.
4. Поддерживаемые БД. Практически во всех SCADA-системах осуществлена поддержка SQL-синтаксиса, не зависящего от типа БД, что позволяет создавать независимые программы для анализа информации и использовать уже имеющееся ПО, ориентированное на обработку данных.
5. Графические возможности. Функционально графические интерфейсы SCADA-систем весьма похожи. В каждой из них существует графический объектно-ориентированный редактор с определенным набором анимационных функций. Используемая векторная графика дает возможность осуществлять широкий набор операций над выбранным объектом, а также быстро обновлять изображение на экране средствами анимации. Крайне важен вопрос о поддержке в рассматриваемых системах стандартных функций GUI (Graphic Users Interface). Поскольку большинство рассматриваемых SCADA-систем работает под управлением Windows, это и определяет тип используемого GUI.

### **Эксплуатационные характеристики**

1. Удобство использования. Сервис, предоставляемый SCADA-системами на этапе разработки ППО, обычно очень развит. Почти все они имеют Windows-подобный пользовательский интерфейс, что во многом

повышает удобство их использования, как в процессе разработки, так и в период эксплуатации прикладной задачи.

2. Наличие и качество поддержки. Возможны следующие уровни поддержки: услуги фирмы-разработчика, обслуживание региональными представителями фирмы-разработчика, взаимодействие с системными интеграторами, русификация программ и документации, горячая линия и решение проблем, связанных с индивидуальными требованиями заказчика и др.

В настоящем учебном пособии были изложены технологии разработки АЛП УД на базе SCADA-системы LabView (National Instruments). Рассмотрим основы создания ППО для сбора, обработки и представления в лабораторных практикумах в среде программирования LabVIEW.

## **2. Основы разработки ППО в среде программирования LabVIEW**

Среда программирования LabView является продукцией компании National Instruments и представляет собой средство разработки ППО, близкое по своей логической структуре к конструкциям языков Си или Бейсик. Однако, LabView в отличие от них использует не текстовый язык программирования, а графический – язык G. Он позволяет создавать программы в виде блок-схем.

LabView имеет обширные библиотеки функций для решения различных задач, таких как ввод/вывод, обработка, анализ и визуализация сигналов; контроль и управление технологическими объектами; статистический анализ и комплексные вычисления; взаимодействие процессов и сетевые технологии ActivX и TCP/IP; поддержка SQL запросов; работа с Internet и др.

Программные приложения, создаваемые в LabVIEW носят название **виртуальных инструментов (VI)**, включают две основные панели:

- **передняя или лицевая панель**, осуществляющая интерактивный интерфейс пользователя и имитирующая панель некоторого пульта управления с размещением на нем различных кнопок, графических индикаторов, диалоговых объектов, средств управления и индикации и т.д.;
- **функциональная панель или блок-схема**, в которой с помощью языка G осуществляется процесс разработки исходного кода виртуального инструмента в виде отдельных графических пиктограмм, осуществляющих различные функции, и связей между ними.

При этом виртуальные инструменты являются также аналогами функций языков программирования и подчиняются принципам иерархичности и модульности. В результате формируемые VI оказываются составленными из VI более низкого уровня (subVI), реализуя при этом концепцию модульного

программирования. Возможно также накапливать и создавать собственные библиотеки виртуальных инструментов.

## **2.1. Основные понятия и панели LabVIEW**

Запуск среды программирования LabVIEW осуществляется либо двойным кликом мыши на ярлыке LabVIEW, который находится на рабочем столе, либо из раздела Программы – National Instruments LabVIEW (изложение для Windows 9x, NT, 2000). При входе в главное меню LabVIEW (версия 5.1) пользователю предлагается создание нового виртуального инструмента (**New VI**) или открытие уже существующего (**Open VI**).

Разработка VI осуществляется на двух панелях, находящихся в двух окнах, - передней и функциональной. На передней панели визуально размещаются средства управления и индикации, на функциональной – составляется блок-схема или исходный код будущего VI. Структура панелей одинакова. Основным элементом каждой панели является рабочая область, снабженная горизонтальным и вертикальным скролингами, в которой и размещаются элементы. Также на панелях имеется верхнее меню и набор функциональных кнопок. Размер окон может регулироваться пользователем. Размещение одновременно двух окон на экране – **Ctrl+T**. Активизация одной из панелей осуществляется посредством клика мыши в ее области или **Ctrl+E**. Имя панели соответствует имени загруженного в него VI. Если VI новый, то панель носит название Untitled. Сохранение VI осуществляется через верхнее меню любой из панелей – **File-Save** или **File-Save As** для сохранения под новым именем.

Для обеих панелей доступна панель **Tools Palette** (рис.1), включающая набор управляющих кнопок для изменения режима редактирования. Перечислим некоторые из них:



**Рис. 1. Панели Tools, Controls и Functions**

- кнопка «указательный палец» – служит для изменения позиций выключателей и кнопок, управления значениями цифровых регуляторов, настройки виртуальных осциллографов и др.
- кнопка «стрелка» – выделение, перемещение объектов, изменение их размера.
- кнопка «А» – открытие и редактирование текстового окна.
- кнопка «катушка» – служит для соединения объектов на функциональной панели.
- кнопка «кисть» – раскрашивание объектов или фона.
- кнопка «рука» – перемещение рабочей области панели в окне.
- кнопка «пипетка» – выбор текущего цвета из имеющихся на панели.
- кнопка «красный круг» – для размещения и снятия точек остановки выполнения программы на функциональной панели.
- кнопка «Р» – для размещения на функциональной панели локальных окон для отображения текущих значений данных, передаваемых в ходе выполнения программы.

При активной передней панели становится доступной панель **Controls** (рис.1). С ее помощью осуществляется визуальное размещение регуляторов и индикаторов на передней панели VI. **Регуляторы** предназначены для ввода информации в ходе выполнения программы, **индикаторы** – для вывода. В панели Controls они распределены по отдельным группам по некоторым признакам – числовые, логические, строковые, массивы, диалоговые, ActivX, Internet и др.

При активировании функциональной панели становится доступной



панель **Functions** (рис.1), которая аналогично панели Controls включает систематизированные наборы стандартных элементов в виде отдельных пиктограмм, из которых осуществляется составление блок-схемы VI.

На передней и функциональной панелях также размещаются управляющие кнопки (рис. 2), такие как



- кнопка «стрелка» – пуск выполнения программы; если в программе имеются

**Рис. 2. Управляющие кнопки** ошибки, то данная кнопка расколота на две части;

- кнопка «стрелки в цикле» – запуск программы в циклическом режиме;
- кнопка «красный круг» – остановка выполнения программы;
- кнопка «две вертикальные черты» – пауза в выполнении программы.

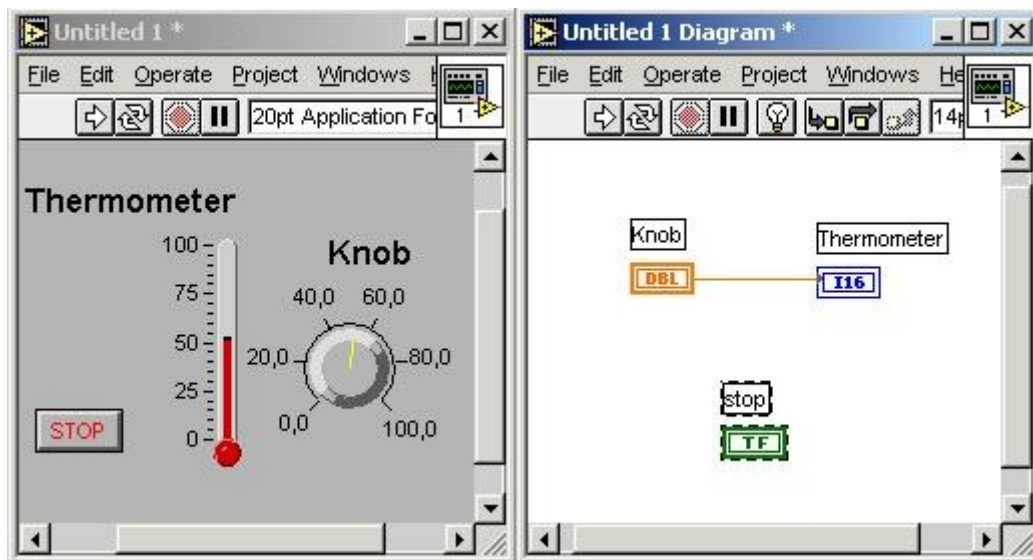
Процесс разработки VI включает:

1 Размещение регуляторов и индикаторов на передней панели VI. Для этого из панели Controls выбирается объект требуемого типа и внешнего вида и размещается в требуемом месте на передней панели. При этом его размер, цвет, описание и название могут в последующем меняться.

2 Добавление требуемых для прикладной задачи структур и функций на функциональной панели. Для этого из панели Functions выбираются соответствующие структуры и функции, пиктограммы которых размещаются на функциональной панели.

3 Соединение регуляторов, индикаторов, констант, функций и др. на функциональной панели при помощи проводки.

**Регуляторы и индикаторы** выполняют те же функции, что и входные и выходные параметры в текстовых языках программирования. При размещении регулятора/индикатора на передней панели, LabView создает соответствующую пиктограмму на блок-схеме. Символы на терминале соответствуют типу данных терминала. Например, **DBL** – терминал представляет данные в виде вещественных чисел с двойной точностью, **TF** – логический терминал, **I16** – терминал 16 – битных целых и др. (рис. 3).



**Рис. 3. Регуляторы и индикаторы**

При нажатии правой кнопки мыши на регуляторе/индикаторе (как на передней, так и функциональной панели) появляется контекстное меню, с помощью которого возможно осуществить:

- замену индикатора на регулятор и наоборот (Change to Control, Change to Indicator);
- быстрый поиск терминала на функциональной панели (Find Terminal) и регулятора/индикатора на передней панели (Find Control, Find Indicator);
- демонстрацию или отказ от нее для названия и описания регулятора/индикатора (Show-Label, Show-Caption);
- настройку параметров регулятора/индикатора (Data Operations);
- замену на другой регулятор/индикатор (Replace);
- получение справки по используемой функции (Online Help);
- открытие для функций соответствующих им констант, индикаторов и регуляторов (Create Constant, Create Indicator, Create Control );
- и др.

**Терминалы** представляют собой области функции, через которые передается информация. Они аналогичны параметрам в текстовых языках программирования. Для того, чтобы увидеть какие терминалы включает данная функция необходима по правой кнопке мыши на пиктограмме из контекстного меню выбрать Show-Terminals (рис. 4).



**Рис. 4. Терминалы**

**Провода** - пути данных между терминалами. Они аналогичны переменным на обычных языках (рис.3). Данные идут в только одном направлении, с исходного терминала на один или более терминалов адресата. Провода имеют различную толщину и цвет. Синий цвет соответствует целым числам, оранжевый – вещественным числам, зеленый – логическим, лиловый – строковым данным. По мере перехода от скаляра к массиву и кластеру увеличивается толщина провода.

Для соединения терминалов необходимо подвести курсор мыши к исходному терминалу (из панели Tools выбрана кнопка «катушка»). При этом отдельные части пиктограммы, соответствующие различным терминалам начинают мигать, а также появляются всплывающие подсказки для облегчения идентификации терминала. После выбора нужного терминала на нем необходимо кликнуть левой кнопкой мыши. В этом случае один конец провода станет закрепленным за данным терминалом. Другой конец, перемещая курсор мыши, необходимо подвести к терминалу адресата и кликнуть левой кнопкой на нем. Если данное соединение возможно, то провод станет соответствующего типу передаваемых данных цвета, в противном случае он станет пунктирным черного цвета. Удаление всех некорректных соединений **Ctrl+B**.

В случае необходимости возможно удаление отдельных сегментов связей, ведение ответвлений от существующих проводов.

**Пиктограмма VI** соответствует каждому виртуальному инструменту и располагается в правом верхнем углу передней панели (рис. 3). Для редактирования пиктограммы используется упрощенный графический редактор, позволяющий создавать изображение, закрашивая его отдельные пиксели. Для этого необходимо вызвать контекстное меню на иконке в правом верхнем углу лицевой панели, и выбрать Edit Icon.

**Коннектор** представляет собой программный интерфейс виртуального инструмента. При использовании регуляторов или индикаторов на передней панели для передачи данных в VI, эти объекты должны иметь терминалы на панели коннектора. Он вызывается из контекстного меню на пиктограмме VI Show Connector. При этом выделяются терминалы для регуляторов на левой половине панели, а для индикаторов – на правой в соответствии с их количеством. Соответствие терминала индикатору или регулятору устанавливается щелчком левой кнопки мыши на терминале коннектора, а затем на соответствующем индикаторе или регуляторе. Это особенно важно при использовании разрабатываемого VI в других виртуальных инструментах для обеспечения возможности его подключения.

**SubVI** является аналогом подпрограммы. В создаваемом VI возможно использование любого виртуального инструмента, имеющего коннектор. Базовые настройки и тип разрабатываемого VI устанавливаются в контекстном меню пиктограммы – пункт VI Setup.

**Панель Controls** служит для добавления регуляторов и индикаторов к передней панели. Если панель Controls не видна на экране, ее можно открыть

через верхнее меню Windows – Show Controls Palette. Панель Controls доступна, только если активно окно передней панели. Рассмотрим основные подпанели панели Controls.

- **Numeric** (числовые значения). Состоит из регуляторов и индикаторов для числовых данных.
- **Boolean** (Булевы значения). Состоит из регуляторов и индикаторов для булевых величин.
- **String&Table** (строковые значения и таблицы). Состоит из регуляторов и индикаторов для ASCII строк и таблиц.
- **List & Ring** (списки и закольцованные списки). Состоит из регуляторов и индикаторов для меню, выполненных в форме списков и закольцованных списков.
- **Array & Cluster** (массивы и кластеры). Состоит из регуляторов и индикаторов для группировки наборов типов данных.
- **Graph** (виртуальные осциллографы). Состоит из индикаторов, для построения графиков данных в графах или диаграммах в реальном масштабе времени.
- **Path & Refnum** (пути и ссылки). Состоит из регуляторов и индикаторов для путей и ссылок.
- **Decorations** (оформление). Состоит из графических объектов для настройки дисплеев передней панели.
- **Select Control** (выбор регулятора). Отображает диалоговое окно для загрузки самодельных элементов управления.
- **User Controls** (средства управления пользователем). Состоит из специальных средств управления, которые формирует сам пользователь.
- **ActiveX** (объекты ActiveX). Состоит из средств управления, позволяющих внедрить объекты ActiveX на переднюю панель.
- **Dialog** (диалоговая панель). Состоит из стандартных объектов для формирования диалога с пользователем.
- **IMAQ Vision** (обработка изображений). Состоит из средств обработки и анализа изображений.
- **Internet Toolkit** (работа с Internet). Состоит из средств управления, располагаемых на передней панели, позволяющих организовывать работу виртуальных инструментов в сети Internet (ftp, электронная почта, telnet, CGI и другие).

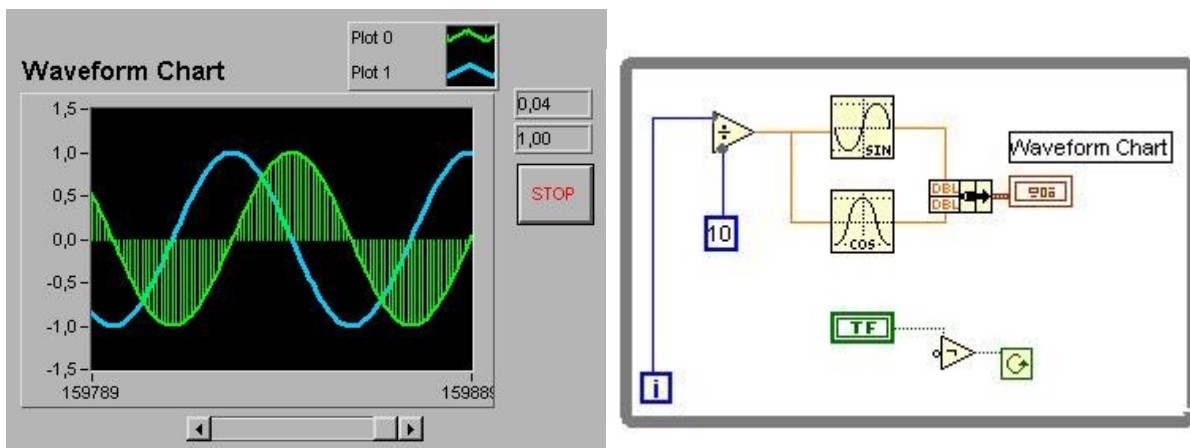
**Панель Functions предназначена** для формирования блок-схемы VI. Каждая пиктограмма на панели открывает подпанель пиктограмм нижнего уровня. Если панель Functions не видна на экране, ее можно открыть через верхнее меню Windows – Show Functions Palette. Панель Functions доступна, только если активно окно функциональной панели. Рассмотрим основные подпанели панели Functions.

- **Structures** (структуры). Состоит из управляющих структур программы, таких как циклы For Loop, While Loop и другие.
- **Numeric** (числовые функции). Состоит из тригонометрических, логарифмических и других функций.
- **Boolean** (Булевы функции). Состоит из логических и Булевых функций.
- **String** (строковые функции). Состоит из функций для работы со строковыми величинами.
- **Array** (массивы). Состоит из функций для обработки массивов.
- **Cluster** (кластеры). Состоит из функций для обработки кластеров.
- **Comparison** (сравнение). Состоит из функций для сравнения переменных.
- **Time & Dialog** (время и диалог). Состоит из функций для диалоговых окон, синхронизации, и обработки ошибок.
- **File I/O** (ввода/вывода файла). Состоит из функций для осуществления операций по вводу/выводу файлов.
- **Instrument I/O** (инструменты ввода/вывода). Состоит из VI для связи и управления приборами различной архитектуры.
- **Instrument Drivers** (драйверы приборов). Состоит из VI, способных управлять внешними приборами, осциллоскопами, генераторами, и т.д., через последовательный порт или интерфейс GPIB.
- **Data Acquisition** (сбор данных). Состоит из VI для использования плат сбора данных.
- **Signal Processing** (обработка сигналов). Состоит из VI для генерации и обработки сигналов.
- **Mathematics** (математические). Состоит из оптимизационных, алгебраических, интегральных, дифференциальных и других функций.
- **Graphics & Sound** (графика и звук). Состоит из VI для работы трехмерной графикой, изображениями и звуком.
- **Communication** (связи). Состоит из виртуальных приборов для работы с сетями TCP, DDE и др.
- **Application Control** (управление приложением). Состоит из VI, управляющих виртуальными приборами.
- **Advanced** (расширенная). Состоит из разных функций типа функции библиотечного запроса, манипуляции данных и др.
- **Report Generation** (генерация отчета). Состоит из VI, используемых для подготовки отчетных документов.
- **Tutorial** (обучающие программы). Состоит из VI, используемых в обучающей программе LabVIEW.
- **User Libraries** (пользовательские библиотеки). С помощью нее организуется быстрый доступ к нужному vi.
- **Select VI** (выбор VI). Состоит из диалогового окна для внедрения подпрограмм в текущий ВП.
- **IMAQ Vision** (обработка изображений). Состоит из VI, используемых для обработки и анализа изображений.

- **Image Acquisition** (получение изображения). Состоит из VI, используемых для получения и обработки изображений.
- **Internet Toolkit** (работа с Internet). Состоит из VI, используемых для работы в сети Internet (ftp, электронная почта, telnet, CGI и другие).
- **SQL** (SQL запросы). Состоит из VI, используемых для организации связи с SQL сервером и обработки запросов.

## 2.2. Использование структур и построение графиков ввода

Для графического отображения полученных данных используются диаграммы или виртуальные осциллографы. Диаграмма (**Chart**) - это виртуальный осциллограф, экран которого обновляется по мере поступления новых данных. Располагается в панели Controls-Graph-Waveform Chart. Настройка диаграммы осуществляется пользователем. При этом могут быть использованы полоса прокрутки (scrollbar), легенда (legend), палитра (palette), цифровой дисплей (digital display) и др. Возможно одновременное отображение на одной диаграмме нескольких зависимостей разным цветом или типом линии, имеющих одну вертикальную шкалу или несколько (контекстное меню на диаграмме Stack Plots/Overlay Plots). Для очистки экрана **осциллографа** необходимо в его контекстном меню выбрать Data Operations-Clear Chart.



**Рис. 5. Диаграмма Chart**(Выводит на график значения sin и cos до нажатия кнопки stop. Используется функция Bundle из панели Cluster)

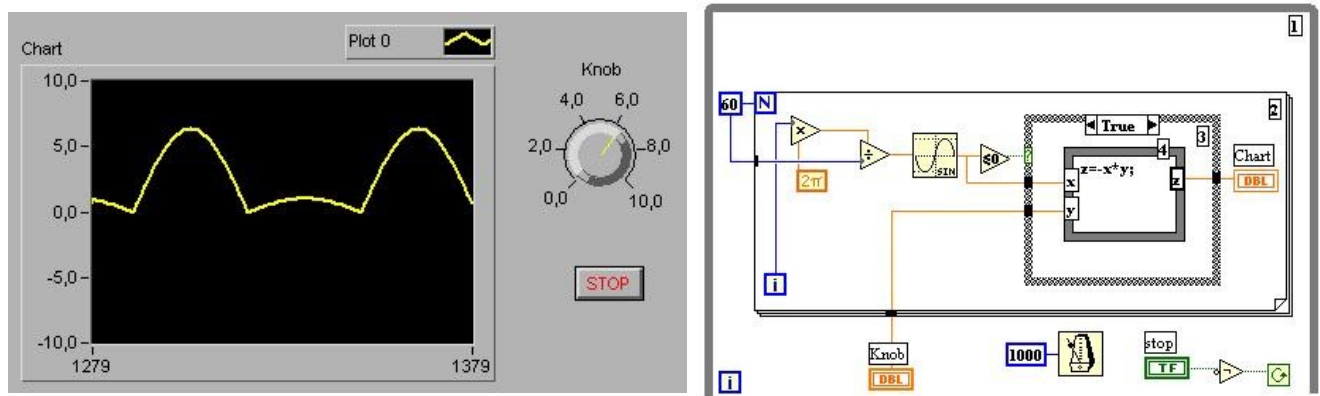
Возможны различные виды представления графиков в виртуальном осциллографе (Data Operations-Update Mode):

- **Strip** - отображение информации подобно действию самописца на бумажной ленте, т.е. новое значение наносится слева, если линия дошла до края области отображения, предыдущие значения начинают сдвигаться вправо.

- Score - отображение информации подобно работе осциллографа, т.е. когда линия достигает правого края экрана, экран обновляется, и линия снова идет с левого края.
- Sweep подобен режиму Score, но экран не очищается при достижении линией правой границы дисплея. Место начала нового цикла отмечает красная вертикальная черта, которая смещается влево по мере поступления новой информации.

**Структура** предназначена для управления прохождением данных в виртуальных инструментах. В языке G используется пять структур.

1. While Loop - условный цикл.
2. For Loop - счетный цикл.
3. Case Structure - выбор.
4. Sequence Structure - последовательность.
5. Formula Node - формульный блок.



**Рис. 6. Структуры While Loop, For, Case, Formula Node**

Условный и счетный циклы (While Loop и For Loop) являются базовыми структурами языка G, как и многих других текстовых языков программирования.

Условный цикл (**While Loop**) осуществляет выполнение части программы определенное число раз, которое задается некоторым условием. Цикл While Loop включает (рис. 6, рамка 1).

- Ограниченную прямоугольную область, изменяемого размера, - тело цикла.
- Терминал условия, определяющий момент окончания работы цикла (момент, когда на него подается значение true). VI проверяет значение этого терминала после выполнения цикла, поэтому такой цикл выполняется, по меньшей мере, один раз.



- Терминал итераций (i), который показывает количество выполнений данного цикла. Если цикл выполнен 1 раз, то значением на этом терминале будет 0.

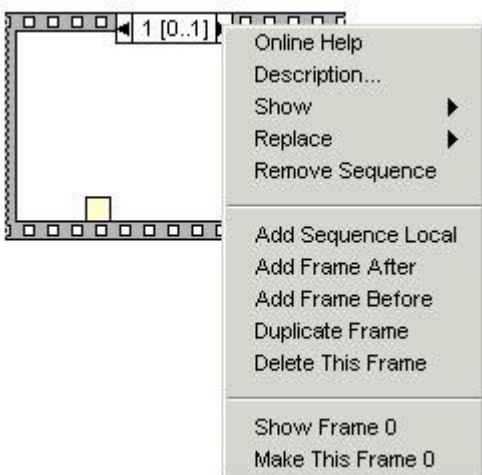
Счетный цикл (**For Loop**) выполняет тело цикла определенное число раз. Цикл For включает (рис. 6, рамка 2).

- Ограниченную прямоугольную область, изменяемого размера, - тело цикла.
- Терминал счетчик. Определяет сколько раз должен выполняться цикл (N).
- Терминал итераций, показывающий текущее число выполненных циклов (i).

В структуре выбор Case (рис. 6, рамка 3) имеется две или более встроенных блок-схемы. Выбор одной из них, которая будет выполнена определяется в зависимости от значения, поданного на вход данной структуры. Структура Case включает.

- Терминал выбора (?). Значение, подаваемое на него, может быть целым, логическим или строковым.
- Переключатель блок-схем (True \ False \ и т.д.). Позволяет переходить от одной блок-схемы к другой. Содержит по умолчанию два окна
- True и False. При необходимости количество блок-схем выбора может быть увеличено. Кроме True и False в качестве значений переключателя могут использоваться целые числа или строковые значения. В данном случае, значение, поданное на терминал выбора, будет сравниваться со значением переключателя данной блок-схемы. Всегда необходимо предусматривать блок-схему для False.

Формульный блок **Formula Node** (рис. 5, рамка 4) позволяет вводить



формулы в обычном виде прямо в блок-схему. Особенно это удобно, когда выражение имеет много переменных и сложный вид. Формулы вводятся как простой текст. При этом создаются терминалы на границе блока (контекстное меню Add Input или Add Output), в которые вписываются имена переменных. Каждое выражение заканчивается разделителем ";". Описание синтаксиса формул, а также используемых функций и операторов содержится в Help-Formula Node. Структура последовательность **Sequence**

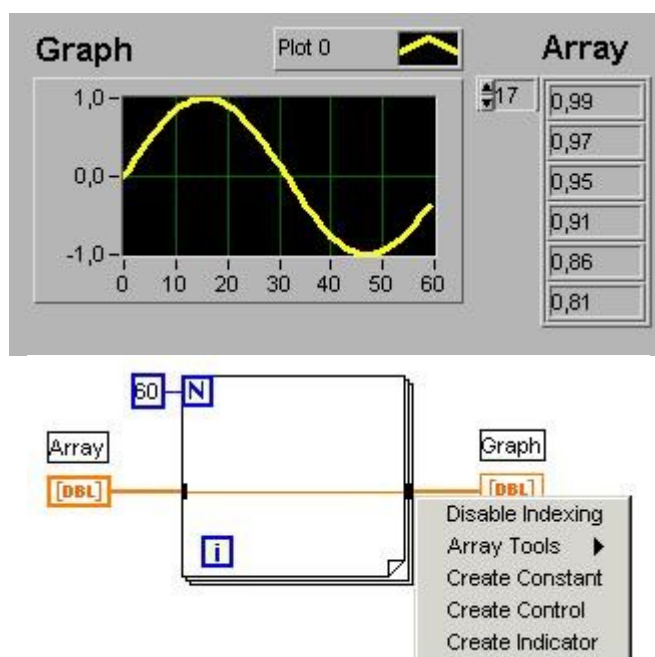
Рис. 7. Структура Sequence



**Structure** (рис. 7) выполняет встроенные в нее блок-схемы последовательно в определенном порядке. Количество встроенных блок-схем определяется числом фреймов данной структуры. Их количество добавляется при помощи контекстного меню - Add Frame After, Add Frame Before. Для передачи значений переменных из фрейма в фрейм используются локальные переменные структуры (контекстное меню - Add Sequence Local variable), создаваемые на границе фрейма. Данные, связанные с такой переменной доступны во всех последующих фреймах и не доступны в предыдущих.

### 2.3. Массивы и кластеры

**Массив** - набор данных одного типа. Массив может иметь одно или несколько измерений. Доступ к элементу массива осуществляется по индексу. Индекс - это число от 0 до n-1, где n это число элементов массива. Для инициализации массива необходимо выбрать в панели Functions-Array-Array Constant или Controls-Array&Cluster. Используя Operating tool из палитры инструментов, вы можете выбрать числовую, логическую или строковую константу, которую нужно поместить в пустой массив.



**Рис. 8. Массивы и диаграмма Graph**

Структуры For Loop и While Loop могут использоваться для автоиндексации массивов (рис.8). Если связать внешний массив с блоком внутри цикла через входной канал, то цикл будет последовательно считывать элементы массива, по одному за цикл. Цикл будет считывать скаляры из

одномерного массива, одномерные массивы из двумерного, и так далее. Если массив связан через выходной канал, то элементы будут записываться в массив.

Для включения и выключения автоиндексации (рис. 8) используется контекстное меню на входном/выходном каналах цикла - маленьких черных квадратов границы цикла (Disable Indexing / Enable Indexing). Для сборки элементов с образованием массива используются функция Build Array (Functions-Build Array).

**Кластеры** - упорядоченная совокупность элементов различного типа. Сборка и разборка кластера осуществляется функциями Functions-Bundle, Functions-Unbundle. Кластеры могут использоваться при выводе нескольких графиков на диаграмме Chart (рис. 5).

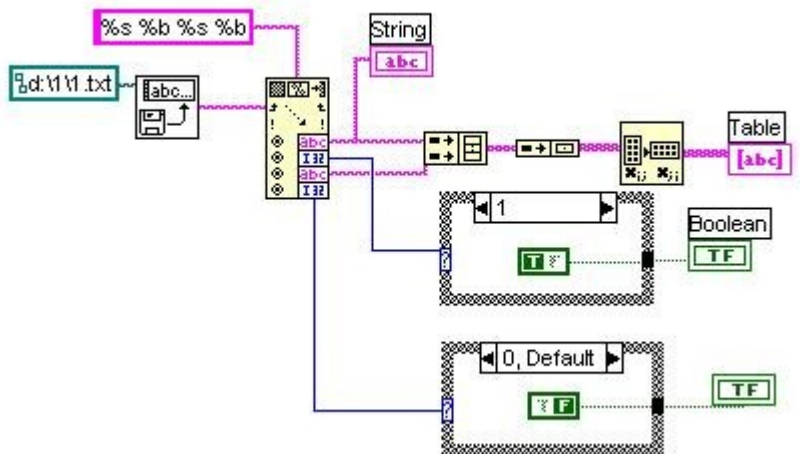
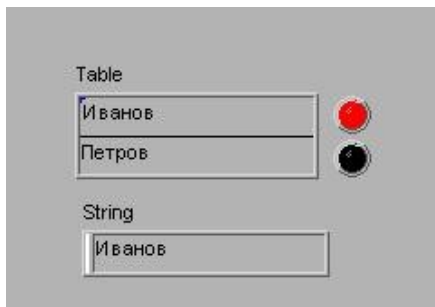
Осциллограф Waveform Graph (рис.8) позволяет наблюдать временные зависимости сигналов. Он регистрирует процесс за время одного пуска программы на числе выборок, которое устанавливается в программе. Он обновляется при новом запуске VI и может быть много лучевым.

#### **2.4. Строки, таблицы, файлы.**

**Строка** представляет собой набор символов ASCII. На передней панели vi могут быть размещены строковые индикаторы и регуляторы (Controls-String&Table). Для работы со строками предусмотрены специальные функции (Functions - String), обеспечивающие объединение строк, определение длины строки, выделение фрагмента строки, редактирование строк, разборку строки на составляющие фрагменты, конвертацию других типов переменных в строковую и др.

**Таблицы** - двумерные массивы строк (Controls-String&Table). Предусмотрены линейки скроллинга, цифровые индикаторы для показа индексов строк и столбцов, отсечение заголовков строк и столбцов и др.

Функции **ввода/вывода информации из файла** (Functions - File I/O) дают возможность осуществлять запись и считывание данных, перемещение и переименование файлов и папок, запись данных в двоичной форме и др. В Lab-VIEW предусмотрено три формата данных: ASCII Byte Stream - текстовые файлы; Datalog Files - данные хранятся в двоичной форме, с возможностью доступа только из приложений, использующих язык G (подобны файлам баз данных); Binary Byte Stream-данные хранятся в двоичном формате.



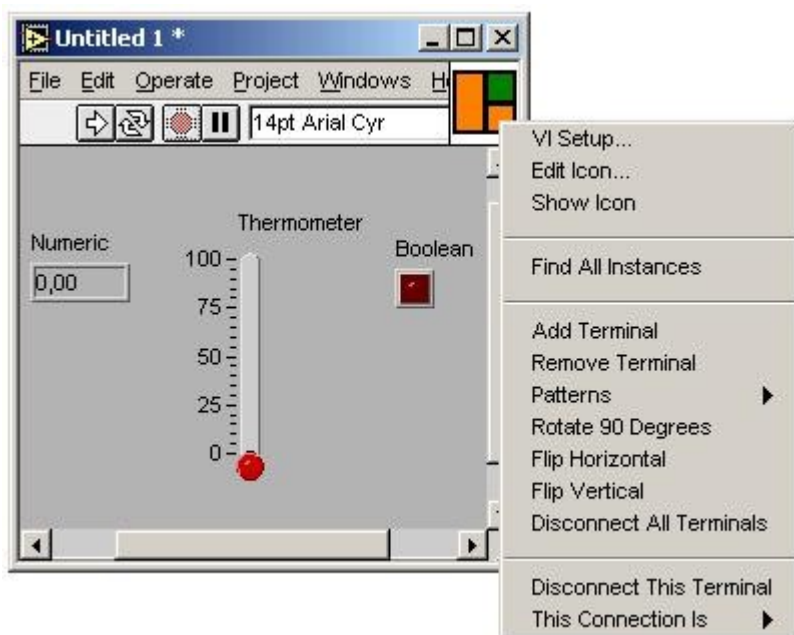
**Рис. 9. Строки, таблицы, файлы**

На рис. 9 приведен пример считывания из текстового файла 1.txt, содержащего четыре строки - Иванов, 1, Петров, 0. Считываемые данные представляются в виде одной строки, которая разбирается в последующем с помощью функции Functions - String - Scan From String. Форматирующая строка задает последовательность и тип данных в сканируемой строке (%s %b %s и т.д.) Строковые данные объединяются в двумерный массив при помощи функции Functions - Array - Build Array, транспонируются и выводятся в таблицу. Числовые (0 или 1) анализируются, переводятся в булевы значения и подаются на индикаторы.

## **2.5. Подпрограммы**

Любой vi может быть использован как подпрограмма при создании в последующем других виртуальных инструментов. Для объединения нескольких функциональных блоков разрабатываемой диаграммы в подпрограмму достаточно выделить их мышкой на диаграмме, удерживая клавишу Shift, и затем выбрать в верхнем меню пункт Edit - Create SubVI. При этом они объединятся в новую подпрограмму с новым значком на функциональной панели. Двойной клик на данном значке позволит вызвать созданную подпрограмму, настроить ее должным образом и сохранить с заданным именем. В последующем данный модуль может быть многократно использован в различных vi.

Для редактирования значка создаваемых vi и подпрограмм достаточно



**Рис. 10. Строки, таблицы, файлы**

в правом верхнем углу и выбрать пункт Show Connector... (рис. 10). При этом пиктограмма разделится на несколько прямоугольников, общий набор и вид которых можно редактировать с помощью всплывающего меню пиктограммы (добавить/удалить терминал - Add Terminal/Remove Terminal, поворот на 90 градусов - Rotate 90 Degrees, другой вид - Patterns... и др.) Для того, что бы сопоставить каждый терминал с определенными данными необходимо ЛКМ кликнуть на нужном терминале, а затем на том индикаторе или регуляторе на передней панели, которой он будет соответствовать. При этом терминал окрасится в цвет соответствующий типу данных указанного индикатора или регулятора. В результате все терминалы будут связаны с определенными входными или выходными данными.

**3. Разработка VI для функционирования в сети Internet**

При разработке лабораторных практикумов удаленного доступа на базе имитационных моделей и реального оборудования возникает проблема дистанционного управления лабораторным стендом и отображения пульта управления на экране обучающегося. Среда программирования LabVIEW предоставляет широкие возможности для создания программного обеспечения по управлению реальными объектами, в том числе с возможностью удаленного доступа через сеть Internet.

Для представления виртуальных инструментов в браузере пользователя используется G Web Server, входящий в состав Internet Developers Toolkit. По-сле инсталляции данного пакета запуск веб сервера осуществляется из верхнего меню Project – Internet Toolkit – Start HTTP Server... При этом запускается виртуальный инструмент, осуществляющий поддержку http сервера на данной ма-шине (рис. 11). После этого к ней можно обращаться

кликнуть ПКМ на пиктограмме vi в правом верхнем углу и выбрать пункт Edit Icon... (рис. 10) С помощью простейших функций графического редактора можно создать собственный вариант иконки.

Настройка входов/выходов (терминалов) подпрограмм осуществляется следующим образом. Необходимо нажать ПКМ на пиктограмме vi

через ее IP или DNS адрес в адресной строке браузера. При отладке программного обеспечения к http серверу можно обращаться локально через localhost.



**Рис. 11. HTTP сервер**

Сервер располагается в папке Program Files/National Instruments/LabVIEW/internet/. В папке home размещаются html страницы. Первоначально загружается файл index.htm, находящийся в папке home. Кроме того, в папке home находится папка cgi-bin, где размещаются виртуальные инструменты, осуществляющие CGI интерфейс.

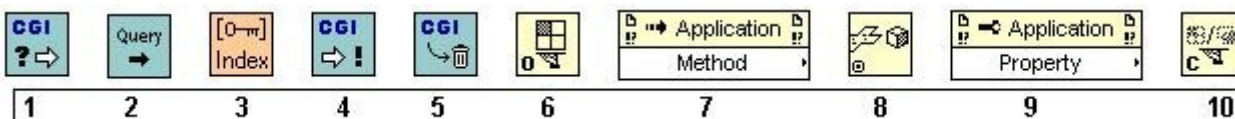
Для размещения изображения передней панели vi на html странице используется следующий тэг:

```
<P><IMG
SRC="http://62.76.177.133/.snap?teplotgen_otd.vi"
ALIGN="BOTTOM" BORDER="1" USEMAP="#panel" ISMAP></P>
```

Возможно вместо параметра snap использовать параметр monitor для анимационных изображений, тогда после имени vi задаются параметры &refresh=1&lifespan=60 (обновление будет через 1 сек в течение 60 сек). Не все браузеры поддерживают данный режим.

Отображение передней панели виртуального инструмента на машине клиента будет возможно только в том случае, если на сервере данный vi уже будет запущен. Для обеспечения возможности удаленного запуска через Internet браузер виртуального инструмента, обслуживающего лабораторную установку, а также последующего управления ею, необходима разработка vi, осуществляющего поддержку CGI интерфейса. Данный vi обязательно должен находиться внутри папки cgi-bin.

Рассмотрим набор основных функций, используемых при разработке vi для cgi интерфейса и управления другими приложениями (рис. 12).



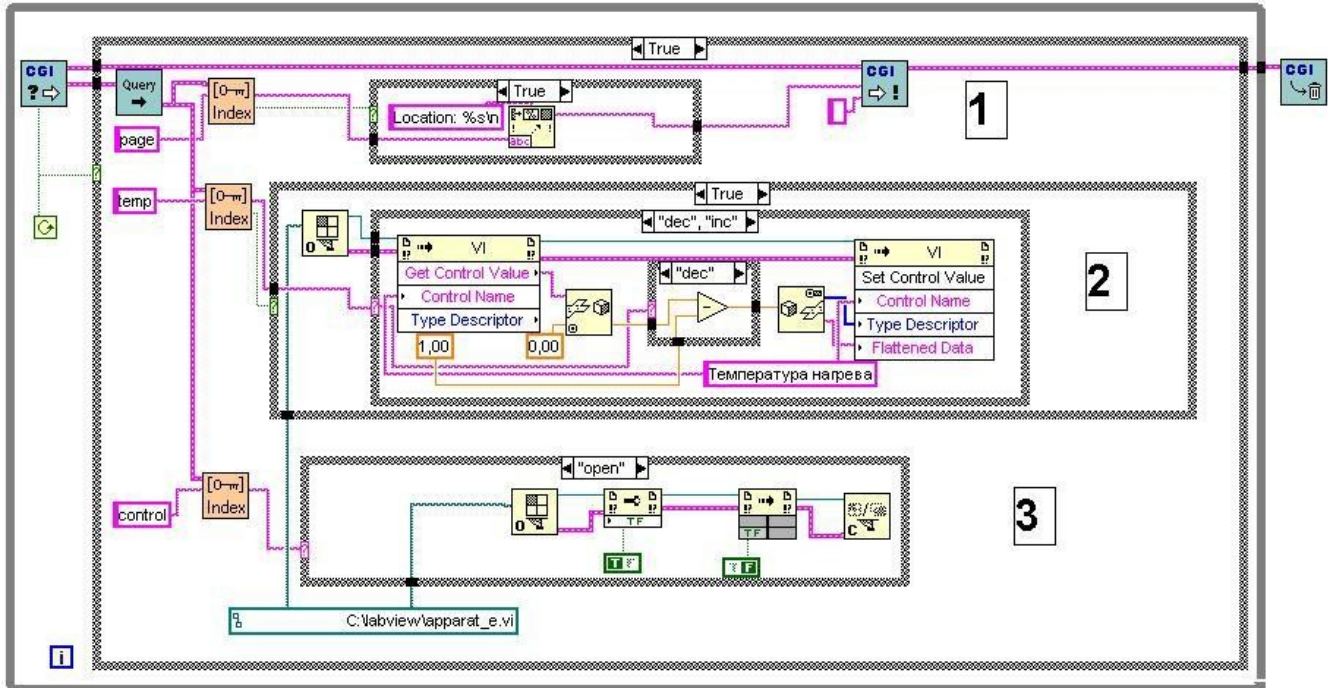
**Рис. 12. Функции для CGI интерфейса и работы с другими приложениями**

1. CGI Reade Request (Internet-CGI-CGI) – ожидает запрос и при подключении использует env - окружение (Keyed Array). Cgi connection inf - используется для передачи информации о соединении.
2. CGI Get Query Parameters (Internet-CGI)- возвращает параметры, пришедшие с cgi запросом (от окружения), которые в последующем и приходится разбирать по переменным.
3. Keyed Array Index (Internet-CGI-Keyed Array)- возвращает значение элемента массива (array in) по ключу -(key) – имени переменной окружения (page – страница которой передается дальнейшее управление, temp– увеличение/уменьшение температуры, control– запуск/закрытие виртуального инструмента). valid key - указывает, что данная переменная получена; value - содержит значение полученной переменной.
4. CGI Write Reply (Internet-CGI-CGI)- пишет ответ на http соединение. В header его подается значение переменной page – страницы, на которую пере-дается управление в последующем.
5. CGI Release (Internet-CGI-CGI)– информирует сервер, что обработка запроса закончена.
6. Open VI Reference (Application Control - Open VI reference) - возвращает ссылку на VI указанный в vi path -- локальный путь к запускаемому vi.
7. Invoke Node (Application Control) - вызов методов и действий с vi.
  - установка работы с виртуальным инструментом – контекстное меню, Select VI Server Class - Virtual Instruments
  - метод - контекстное меню - Methods -
    - Get Control Value - получение значения от указанного vi по соответствующей переменной (по Control Name). Переменная указывается четко по ее названию в запущенном vi (можно русское название). Соединяются между собой через reference и ошибки.
    - Set Control Value - установка (передача) значения установленной переменной.
    - Run VI - запускает vi, необходимо подать False, чтобы не ждал завершения выполнения программы.
    - Abort VI - останавливает работу запущенного vi.
8. Unflatten From String / Flatten To String (Advanced - Data Manipulation) - конвертирует бинарную строку к приведенному типу, какой указывает пользователь (в вещественные, булевы и др.), или нечто (вещественные, булевы и др.) конвертирует в бинарную строку для передачи запущенному vi - через Set Control Value (Type Description - строка).
9. Property Node (Application Control) - пишет или считывает информацию о свойствах vi.



- открытие передней панели - контекстное меню-Property-Front Panel Window-Open (true - открыть, false - закрыть), затем контекстное меню-Change to Write. Соединяется через reference.

1. Close Application or VI Reference (Application Control - Open VI reference) - закрывает открытые vi или соединения.



**Рис. 13. Пример виртуального инструмента для CGI интерфейса.**

На рис. 13 изображен вариант виртуального инструмента, который позволяет осуществлять дистанционный запуск и управление виртуальным инструментом `apparat_e.vi`. В качестве входных параметров могут быть заданы:

- название Internet страницы, которая в последующем будет загружена клиенту (блок 1) – параметр `page`;
- команда на запуск двигателя (`dvig`), насоса (`nasos`), нагревателя (`pagrev`) или уменьшение/увеличение параметра регулирования (`dec`, `inc` именно данный вариант показан на рис. 13) (блок 2) – параметр `temp`;
- открытие или закрытие vi на сервере (блок 3) – параметр `control`.

Ссылка на данный vi из запускаемой Internet страницы имеет вид:

```
<a href="http://localhost/cgi-bin/apparat/apparat_cgi_e.vi?control=open&page=/apparat/apparat_map.htm">
```

Текст Internet страницы, на которой отображается передняя панель пульта управления vi apparat\_e.vi может быть представлена в виде:

```
<HTML>
<HEAD>
    <META                HTTP-EQUIV="Content-Type"
CONTENT="text/html;CHARSET=win-1251">
    <META NAME="Author" Content="Internet Toolkit">
</HEAD>

<BODY>    <p align="center"><font color="#800000"
size="4"><strong>Опрос каналов и управление работой
установки</strong></font></p>
<P><IMG SRC="http://localhost/.snap?apparat_e.vi"
ALIGN="BOTTOM" BORDER="1" USEMAP="#panel" ISMAP></P>
<MAP Name="panel">
<AREA Shape="Rect" coords = "408,18,680,162"
        HREF="http://localhost/cgi-
bin/apparat/apparat_cgi_e.vi?page=/apparat/apparat_map.
htm">
<AREA Shape="Rect" coords = "408,182,680,326"
        HREF="http://localhost/cgi-
bin/apparat/apparat_cgi_e.vi?page=/apparat/apparat_map.
htm">
<AREA Shape="Rect" coords = "408,344,680,489"
        HREF="http://localhost/cgi-
bin/apparat/apparat_cgi_e.vi?page=/apparat/apparat_map.
htm">
<AREA Shape="Rect" coords = "280,453,386,483"
        HREF="http://localhost/cgi-
bin/apparat/apparat_cgi_e.vi?control=close&page=/index.
htm">
<AREA Shape="Rect" coords = "58,408,80,441"
        HREF="http://localhost/cgi-
bin/apparat/apparat_cgi_e.vi?page=/apparat/apparat_map.
htm& temp=dvig">

<AREA Shape="Rect" coords = "196,408,218,441"
HREF="http://localhost/cgi-
bin/apparat/apparat_cgi_e.vi?page=/apparat/apparat_map.
htm& temp=nagrev">

<AREA Shape="Rect" coords = "328,408,350,441"
HREF="http://localhost/cgi-
```



```
bin/apparat/apparat_cgi_e.vi?page=/apparat/apparat_map.  
htm& temp=nasos">
```

```
</MAP>
```

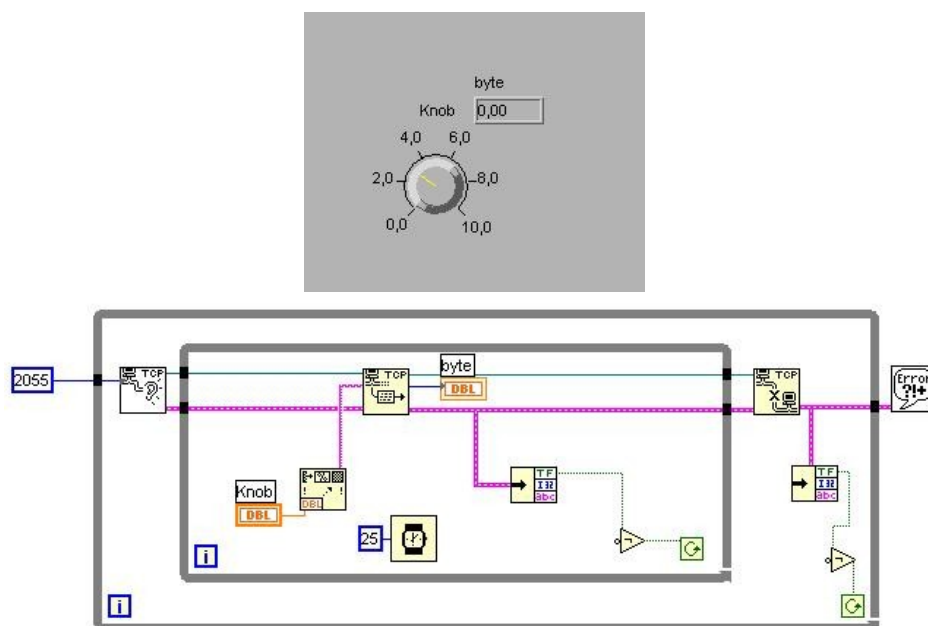
```
</BODY>
```

```
</HTML>
```

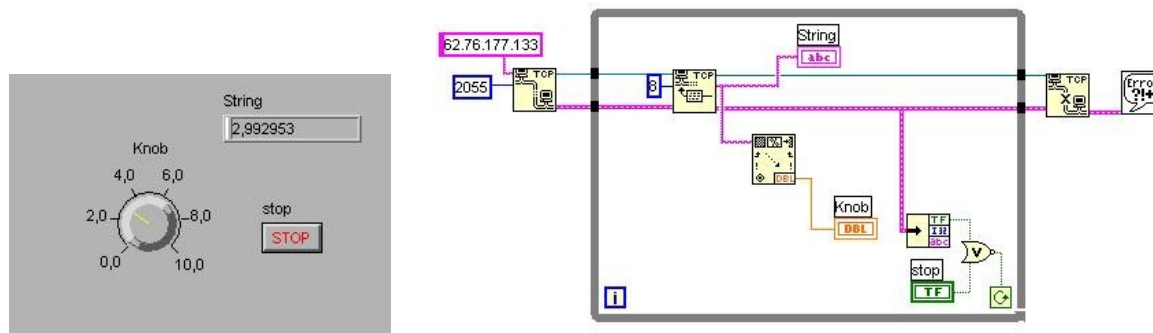
В блоке MAP Name="panel" осуществляется картирование изображения передней панели vi, при помощи которого выделяются органы управления на пульте (кнопки, переключатели, регуляторы, экраны обновления и др.) Нажатие левой кнопки мыши на выделенные фрагменты рисунка вызывает передачу cgi файлу необходимых параметров, в соответствии с которыми он осуществляет определенные действия (открытие соответствующей Internet страницы, запуск двигателя, изменение величины параметра регулирования, открытие vi и др.)

#### 4. Взаимодействие vi по протоколу TCP

Среда программирования LabVIEW позволяет создавать виртуальные инструменты, осуществляющие передачу данных между собой с использованием протокола TCP. Функции для создания таких приложений находятся в панели Functions–Communication–TCP. На рис. 14 и 15 приведен пример двух взаимосвязанных приложений.



**Рис. 14. Взаимодействие по протоколу TCP (серверная часть)**



**Рис. 15. Взаимодействие по протоколу TCP (клиентская часть)**

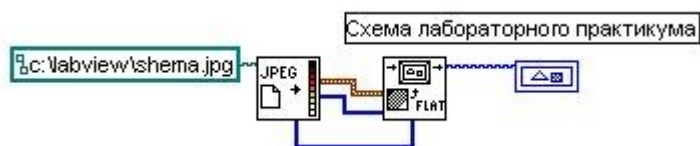
Первый vi выполняет роль сервера, который ожидает подключения по одному из портов TCP (в данном случае взят порт 2055). Для этого использована функция TCP Listen. Когда клиент подключается к серверу с использованием функции TCP Open Connection (указывается IP адрес машины и номер порта), сервер передает (функция TCP Write) по созданному соединению данные (в данном случае это значение цифрового регулятора Knob). Клиент считывает (функция TCP Read) поступившие данные (в количестве 8 байт) и отображает их на своем цифровом индикаторе. При возникновении ошибки или остановке клиента соединение разрывается (функция TCP Close Connection).

Создание подобных приложений также может быть использовано при организации удаленного доступа к лабораторным практикумам как имитационного характера, так и на базе реального оборудования. Рассмотрим способы работы с реальными объектами в среде программирования LabVIEW.

## **5. Использование изображений**

При использовании изображений, содержащихся в графических файлах (jpeg, png, bmp) на передней панели vi размещается индикатор Picture, расположенный на панели Controls–Graph–Picture. В данном случае используемый рисунок будет выводиться на экран в выделенную данным индикатором область. При необходимости на нее могут дополнительно устанавливаться требуемые органы управления и контроля.

Для считывания графического файла, например jpeg, используется функция Functions–Graphics&Sound–Graphics Formats–Read JPEG file (рис. 16).



**Рис. 16. Подключение графических файлов**

На вход ей подается путь к выводимому файлу. На выходе функции формируется карта пиксел, в последующем преобразуемая в «картинку» с использованием функции Functions–Graphics&Sound–Picture Functions–Draw Flattened Pixmap. Сформированное изображение подается на индикатор Picture.

## **6. Сбор данных с реального объекта**

В настоящее время использование компьютеров в научных исследованиях не ограничивается имитационным моделированием на основе математических моделей. Все чаще современная вычислительная техника применяется для приема, обработки и анализа сигналов от реальных физических объектов и управления ими [2]. При этом возникает потребность в электрических датчиках, преобразователях сигналов и специальном программном обеспечении. Среда программирования LabVIEW является удобным программно-аппаратным комплексом для разработки приложений, позволяющих осуществлять опрос датчиков, установленных на объекте исследования, обработку полученной информации, генерацию сигналов для его управления в диапазоне 10 В (от –5В до +5 В для знакопеременных сигналов; от 0 до 10 В для однополярных).

При обработке сигналов и их преобразовании из аналоговых в цифровые и наоборот могут быть использованы платы ЦАП/АЦП. Рассмотрим основные приемы сбора и обработки сигналов с помощью среды программирования Lab-VIEW на примере многофункциональной платы компании National Instruments PCI-MIO-16E-1. Плата устанавливается в свободный слот компьютера и при помощи соединительного кабеля (TYPE SH68-68-EP) подключается к коннектору (ТВХ-68). Датчики, установленные на объекте исследования, подключаются к компьютеру через коннектор в соответствии с настройкой каналов. В таблице 1 приведено соответствие основных каналов клеммам коннектора.

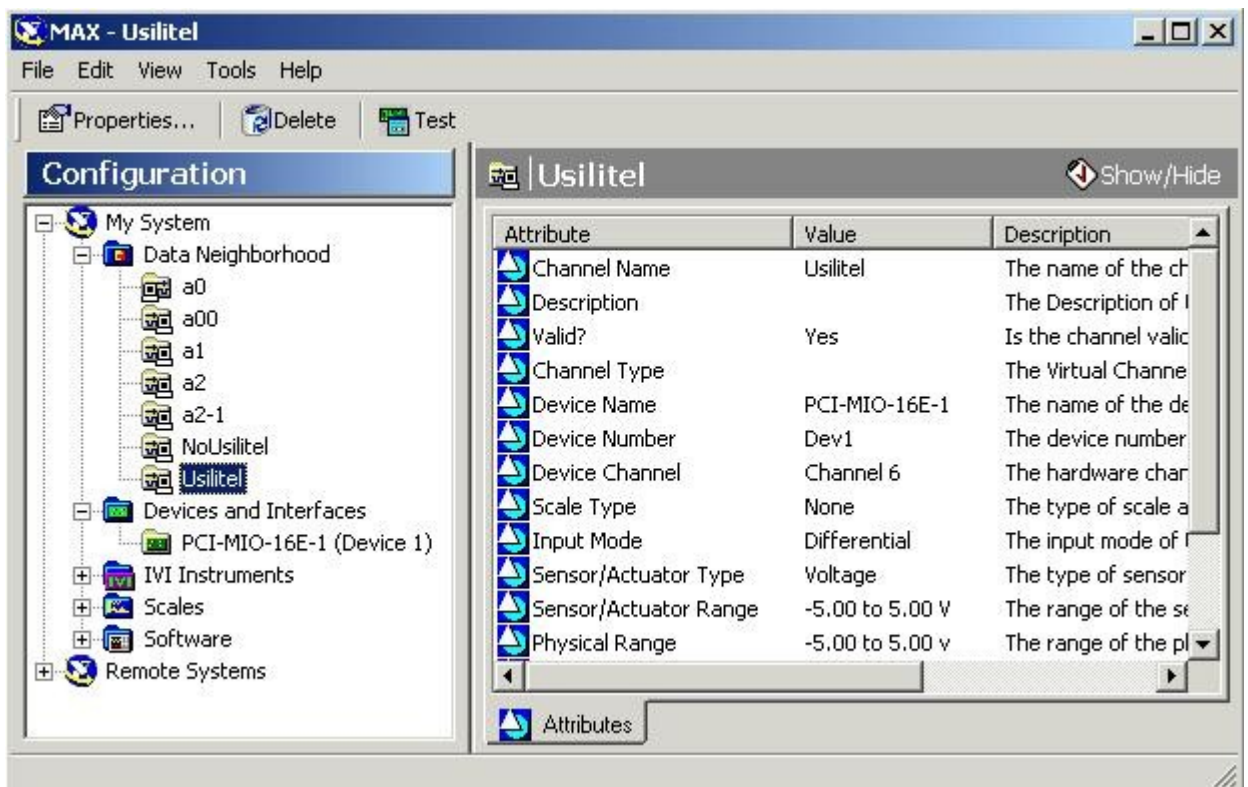
Для настройки каналов ЦАП/АЦП необходимо запустить приложение Measurement & Automation Explorer (рис. 17) и убедиться, что плата установлена и определена верно. При этом в папке Devices and Interfaces появится ЦАП/АЦП, а в скобках будет указан номер данного устройства (например, Device 1). В папке Data Neighborhood содержится список настроенных каналов. Выделив ЛКМ интересующий канал, можно запустить его тест (кнопка Test), просмотреть свойства (кнопка Properties...) или

удалить его (кнопка Delete). Для настройки нового канала необходимо кликнуть ПКМ на папке Data Neighborhood, выбрать пункт Create New..., пункт Virtual Channel и затем нажать кнопку Finish. Далее будет предложено выбрать тип настраиваемого канала: аналоговый вход (Analog Input); аналоговый выход (Analog Output); цифровой вход/выход (Digital I/O).

**Таблица 1**

**Настройка каналов платы PCI-MIO-16E-1 для коннектора TBX-68**

<b>Аналоговый вход АСН &lt;0...15&gt;</b>		<b>Аналоговый выход</b>	
АСН 0	68	DAC0OUT	22
АСН 1	33	DAC1OUT	21
АСН 2	65	<b>Цифровой вход/выход DIO &lt;0...7&gt;</b>	
АСН 3	30	DIO 0	52
АСН 4	28	DIO 1	17
АСН 5	60	DIO 2	49
АСН 6	25	DIO 3	47
АСН 7	57	DIO 4	19
АСН 8	34	DIO 5	51
АСН 9	66	DIO 6	16
АСН 10	31	DIO 7	48
АСН 11	63	<b>Аналоговый вход заземление</b>	24, 27, 29, 32, 56, 59, 64, 67
АСН 12	61		
АСН 13	26	<b>Аналоговый выход заземление</b>	54,55
АСН 14	58		
АСН 15	23	<b>Цифровой вход/выход заземление</b>	4,7,12,13,15,18,35,36,39,44,50,53
<b>Общая незаземленная клемма</b>	62		



**Рис. 17. Приложение Measurement & Automation Explorer**

В следующем окне необходимо указать имя канала (Channel Name) и краткое его описание (Channel Description). Для Analog Input и Analog Output указываются далее:

- тип датчика (напряжение, температура, частота и др.);
- единицы измерения (Units) и диапазон измерения (Range);
- возможное масштабирование (Scaling);
- устройство, для которого данный канал настраивается (What DAQ hardware will be used?);
- номер канала и его соответствие по таблице 1 (Which channel on your DAQ hardware? и Pins);
- тип Analog Input (Which analog input mode will be used?)
  - дифференциальный (Differential);
  - с общим проводом, заземленным в конце (Referenced Single Ended);
  - с общим проводом, незаземленным в конце (Nonreferenced Single Ended).

Для Digital I/O указываются:

- тип цифрового входа/выхода (Read from Port, Read from Line, Write to Port, Write to Line);
- устройство, порт и номер линии с соответствием по таблице 1;

- линии, по которым осуществляется инвертирование цифрового сигнала (Invert Line).

После настройки канала его можно использовать при создании виртуальных инструментов. Функции для получения и формирования аналоговых и цифровых сигналов располагаются в панели Function–Data Acquisition. Перечислим основные из них:

**Аналоговый вход** (указываются номер устройства, имя одного или нескольких каналов):

- однократное считывание по одному из каналов (AI Sample Channel) – выдает значение полученного сигнала по одному каналу в виде числа;
- однократное считывание из группы каналов (AI Sample Channels) – выдает значения полученных сигналов по группе каналов в виде одномерного массива чисел;
- синхронное считывание по одному из каналов (AI Acquire Waveform) – выдает значения полученного сигнала по одному каналу в виде одномерного массива чисел (указывается число выборок и частота сканирования за секунду (1/с));
- синхронное считывание из группы каналов (AI Acquire Waveforms) выдает значения полученных сигналов по группе каналов в виде двумерного массива чисел (указывается число выборок и частота сканирования (1/с)).

Для изменения диапазона измерения используются терминалы high limit и low limit, которые автоматически изменяют коэффициент усиления сигнала (таблица 2).

**Таблица 2**

**Коэффициенты усиления**

<b>Диапазон сигнала</b>	<b>Коэффициент усиления</b>
$\pm 10 \text{ V}$	0.5
$\pm 5 \text{ V}$	1
$\pm 2.5 \text{ V}$	2
$\pm 1 \text{ V}$	5
$\pm 500 \text{ mV}$	10
$\pm 250 \text{ mV}$	20
$\pm 100 \text{ mV}$	50
$\pm 50 \text{ mV}$	100

**Аналоговый выход.** Набор функций подобен аналоговому входу



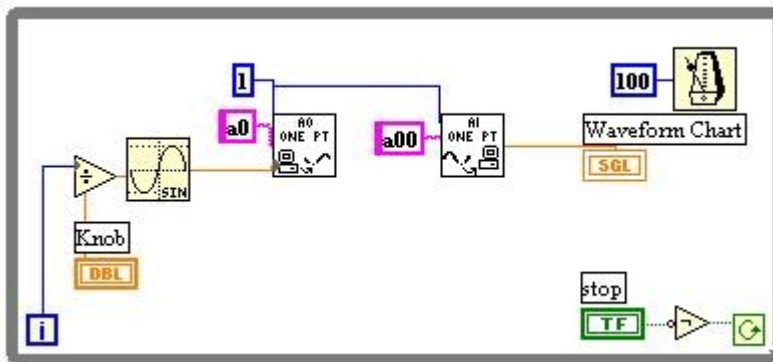
(соответственно – AO Update Channel, AO Update Channels, AO Generate Waveform, AO Generate Waveforms). Также указывается номер устройства и имя канала (каналов). При их выполнении на соответствующие каналы подается заданное выходное напряжение в виде одного числа (AO Update Channel, AO Update Channels) или массива чисел (AO Generate Waveform, AO Generate Waveforms).

**Цифровой вход/выход.** Используются для считывания или записи логического выражения отдельных цифровых линий или цифрового канала в целом. Для считывания/записи логического выражения (true/false) отдельной цифровой линии настроенного канала используются функции Read from Digital Line и Write to Digital Line. Для них указываются номер устройства и имя цифрового канала, а также номер линии. При выводе данных задается подаваемое логическое значение (true/false), а при получении данных осуществляется его считывание. Для работы с каналом в целом (Read from Digital Port, Write to Digital port) ввод/вывод данных осуществляется аналогично отдельной цифровой линии, но данные имеют вид 8 битного шаблона.

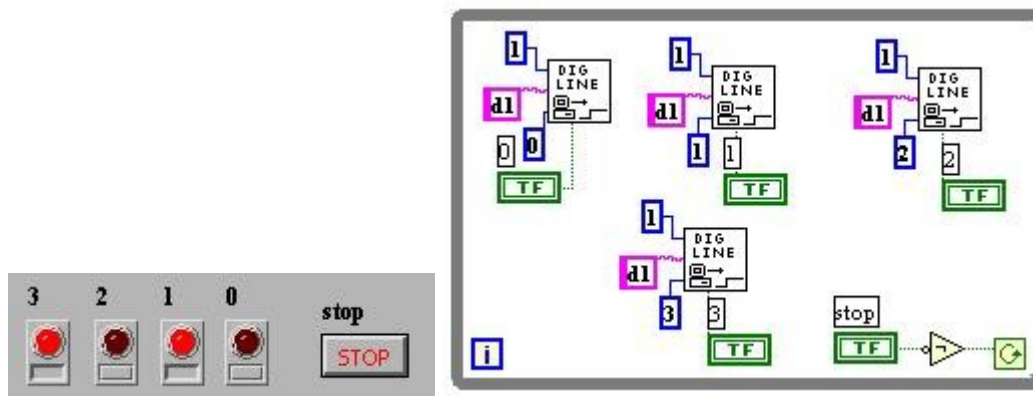
Помимо перечисленных простейших функций для работы каналами ввода/вывода данных в LabVIEW содержится широкий спектр специальных функций, позволяющих осуществлять конфигурирование, сканирование, буферизированный ввод/вывод, очистку и множество других функций. Необходимость их использования при создании виртуальных инструментов определяется конкретной задачей сбора данных.

На рис. 18 приведен пример использования каналов аналогового входа/выхода. Первоначально были настроены каналы a0, как аналоговый выход (DAC0OUT), и a00, как аналоговый вход (канал 1 АСН1). В примере генерируется функция  $\sin$  переменного периода, значения которой подаются на аналоговый выход (a00), соединенный напрямую с аналоговым входом (a0). Коммутация каналов осуществляется на базе устройства DAQ Signal Generator, предназначенного для лабораторного изучения основных подходов обработки сигналов с использованием аппаратного обеспечения компании National Instruments и среды программирования LabVIEW. В Chart выводится сигнал, снимаемый с аналогового входа (a0).





**Рис. 18. Аналоговый вход/выход**



**Рис. 19. Цифровой вход/выход**

На рис. 19 приведен пример использования каналов цифрового входа/выхода. На устройстве DAQ Signal Generator имеется 4 светодиода, подключенных к цифровому порту. Канал d1 настроен как цифровой вход/выход для записи в порт с инверсией сигнала по всем линиям (DIO). Нажатие кнопки на передней панели  $v_i$  вызывает включение/выключение соответствующего ей светодиода на приборе.

В заключении отметим, что изложенный в данном пособии материал по среде программирования LabVIEW охватывает лишь необходимые начальные сведения по созданию виртуальных приборов и их применению в разработке автоматизированных лабораторных практикумов удаленного доступа на базе имитационных математических моделей и реального оборудования. Более полную информацию читатель может получить из справки LabVIEW Help, на серверах [www.ni.com](http://www.ni.com), [www.insysltd.ru](http://www.insysltd.ru), [acs.levsha.ru](http://acs.levsha.ru), [www.vitec.ru](http://www.vitec.ru), [phys.kemsu.ru](http://phys.kemsu.ru) и др.

## **7. Упражнения и лабораторные работы**

1. Разработать виртуальный инструмент, включающий два цифровых регулятора и один цифровой индикатор, позволяющий выполнять сложение



- и умножение двух чисел.
2. Разработать виртуальный инструмент для измерения температуры и уровня жидкости в аппарате. Изменение параметров имитировать с помощью датчика случайных чисел. Предусмотреть как графическое отображение параметров, так и цифровое.  
 # датчик случайных чисел – Functions–Numeric–Random number (0–1)
3. Разработать виртуальный инструмент для измерения температуры жидкости.  
 # имитация датчика температуры – Functions–Tutorial–Digital Thermometer;  
 # логическое отрицание – Functions–Boolean–Not;  
 # временная синхронизация – Functions–Time&Dialog–Wait Until Next ms Multiple.
4. Построить график функции  $y = ax^2 + bx + ab$  в диапазоне от -10 до 20 с шагом 2. Предусмотреть возможность изменения параметров  $a$  и  $b$ .
5. Разработать виртуальный инструмент для управления наполнением емкости жидкостью при помощи насоса. Предусмотреть возможность регулирования расхода жидкости, сигнализацию заполнения емкости (достижение уровня – 0.95) и автоматическое отключение насоса.
6. Разработать виртуальный инструмент, в котором осуществляется наполнение емкости насосом (аналогично работе 3), затем 5 секундная задержка на физико-химические превращения (горит сигнальный индикатор) и далее опорожнение емкости.  
 # временная задержка – Functions–Time&Dialog–Wait Until Next ms Multiple.  
 # создание локальной переменной – контекстное меню на индикаторе или регуляторе Create–Local Variable (в нее можно писать или из нее читать – контекстное меню на ней Change To Read Local / Change To Write Local).
7. Разработать виртуальный инструмент, отображающий значения функций  $\sin$  и  $\cos$  на одном Waveform Graph за 20 проходов цикла For.
8. Связать и затем развязать кластер, включающий строку (ФИО), массив числовой (оценки экзаменам за сессию), лампочку (горит – переведен на другой семестр, не горит – не переведен).
9. Записать в текстовый файл строку из 10 символов (название функции), 20 числовых значений (значения функции), текущее время, единицу и ноль (для отображения данных значений на двух лампочках). Затем считать данные значения из файла и отобразить их на соответствующих приборах.
10. Разработать виртуальный инструмент для поддержки CGI интерфейса, позволяющий осуществлять дистанционный запуск vi на другом компьютере, его закрытие, изменение значения его числового (увеличение/уменьшение значения переменной на ) и булевого параметров. Продемонстрировать его работу на базе созданной Internet страницы.
11. Используя DAQ Signal Generator разработать виртуальный инструмент,

позволяющий регистрировать значения датчика температуры, включение/выключение и регулировку интенсивности накала светодиода.

12. Разработать виртуальные инструменты, позволяющие осуществлять передачу данных (включение/выключение тумблера и изменение значения цифрового регулятора) от одного к другому с использованием протокола TCP.

## 7. ПЕРЕЧЕНЬ ПРОГРАММНЫХ ПРОДУКТОВ.

При выполнении лабораторных работ по дисциплине ИСПУ используются следующие программные продукты:

1. Trace Mode 5.
2. LabVIEW 6.

Описание использования данных программных продуктов представлено в следующей литературе:

1. Справочное руководство Trace Mode 5.
2. Куцевич Н.А. SCADA-системы. Взгляд со стороны. ([http://www.ipu.ru/period/asu/Contents/Number1/Contents/page\\_22-28.htm](http://www.ipu.ru/period/asu/Contents/Number1/Contents/page_22-28.htm)).
3. Жарков Ф.П., Каратаев В.В., Никифоров В.Ф., Панов В.С. Использование виртуальных инструментов LabVIEW – М.: Солон-Р, Радио и связь, Горячая линия – Телеком, 1999.–268 с.

## **8. КОМПЛЕКТЫ ЗАДАНИЙ ДЛЯ ЛАБОРАТОРНЫХ РАБОТ.**

Задания для выполнения лабораторных работ представлены в методических пособиях по лабораторным работам (см. п.6 по УМКД) и ежегодно обновляются преподавателем.

## 9. ФОНД ТЕСТОВЫХ И КОНТРОЛЬНЫХ ЗАДАНИЙ ДЛЯ ОЦЕНКИ КАЧЕСТВА ЗНАНИЙ ПО ДИСЦИПЛИНЕ.

### Раздел 1. Технические средства АСУ ТП

1. Коммуникационные модули контроллера предназначены для
  - 1) ввода-вывода дискретных и аналоговых сигналов;
  - 2) подключения контроллера к промышленным сетям и организации связи по PtP интерфейсу;
  - 3) решения задач автоматического регулирования, позиционирования и т.д.
  - 4) подключения к базовому блоку контроллера стоек расширения.
2. Сигнальные модули контроллера предназначены для
  - 1) ввода-вывода дискретных и аналоговых сигналов;
  - 2) подключения контроллера к промышленным сетям и организации связи по PtP интерфейсу;
  - 3) решения задач автоматического регулирования, позиционирования и т.д.
  - 4) подключения к базовому блоку контроллера стоек расширения.
3. Интерфейсные модули контроллера предназначены для
  - 1) ввода-вывода дискретных и аналоговых сигналов;
  - 2) подключения контроллера к промышленным сетям и организации связи по PtP интерфейсу;
  - 3) решения задач автоматического регулирования, позиционирования и т.д.
  - 4) подключения к базовому блоку контроллера стоек расширения.
4. Функциональные модули контроллера предназначены для
  - 1) ввода-вывода дискретных и аналоговых сигналов;
  - 2) подключения контроллера к промышленным сетям и организации связи по PtP интерфейсу;
  - 3) решения задач автоматического регулирования, позиционирования и т.д.
  - 4) подключения к базовому блоку контроллера стоек расширения.
5. Какие модули УСО не существуют?
  - 1) Аналоговые.
  - 2) Дискретные.
  - 3) Числоимпульсные.
  - 4) Цифровые.
6. Концевой выключатель электропривода является
  - 1) измерительным преобразователем положения электропривода;
  - 2) датчиком состояния электропривода;
  - 3) регистрирующим прибором для регистрации положения электропривода.
  - 4) Всем перечисленным выше.

7. Регулирующий клапан имеет электрический привод. Каким образом можно организовать управление им в составе непрерывного (например, ПИД-) регулятора?

- 1) Это невозможно, так как электропривод клапана для контроллера – трехпозиционная нагрузка.
- 2) С использованием управляемых преобразователей, например, преобразователя частоты, и выходного аналогового модуля контроллера.
- 3) С использованием в контроллере алгоритмов широтно- и частотно-импульсной модуляции и дискретных выходов контроллера.

8. Регулирующий клапан имеет пневматический привод. Каким образом можно организовать управление им в составе непрерывного (например, ПИД-) регулятора?

- 1) Это невозможно, так как электропривод клапана для контроллера – трехпозиционная нагрузка;
- 2) С использованием электропневматического преобразователя и выходного аналогового модуля контроллера.
- 3) С использованием в контроллере алгоритмов широтно- и частотно-импульсной модуляции и дискретных выходов контроллера.

9. Минимальное число дискретных входов и выходов контроллера, задействованных при управлении электроприводом регулирующего клапана составляет

- 1) два входа, один выход;
- 2) один вход, два выхода;
- 3) два входа, два выхода;
- 4) один вход, один выход.

10. Уровень воды в баке поддерживается с воздействием электромагнитного клапана на сливе. Какой тип нагрузки представляет собой электромагнит клапана для регулятора уровня?

- 1) Трехпозиционная нагрузка.
- 2) Двухпозиционная нагрузка типа «холодильник».
- 3) Двухпозиционная нагрузка типа «нагреватель».

11. Каким образом в преобразователь частоты Hitachi SJ 100 может быть заведен сигнал задания по частоте?

- 1) Через аналоговый токовый вход.
- 2) Через аналоговый вход по напряжению.
- 3) Через дискретные входы.
- 4) Все вышеприведенные возможности имеют место.

12. При организации связи с помощью сигнала по напряжению требуется, чтобы

- 1) сопротивления линии связи и нагрузки (приемника сигнала) были малы;
- 2) сопротивление линии связи было мало, а сопротивление нагрузки – велико;
- 3) сопротивления линии связи и нагрузки были сопоставимы.

13. При организации связи с помощью токового сигнала требуется, чтобы

- 1) суммарное сопротивление линии связи и нагрузки (приемника сигнала) не превышали определенной величины;
- 2) суммарное сопротивление линии связи и нагрузки были не менее определенной величины;
- 3) сопротивления линии связи и нагрузки были сопоставимы.

14. Какой из стандартных токовых сигналов использовать предпочтительнее с точки зрения максимума информации, которую можно из него извлечь?

- 1) 0-20 мА.
- 2) 4-20 мА.
- 3) 0-5 мА.

15. Пассивным выходом модуля УСО является выход,

- 1) требующий внешнего питания;
- 2) требующий процедуры инициализации перед проведением необходимых преобразований;
- 3) требующий для своей работы дополнительно вставляемую в разъем специальную плату.

16. Для гальванической развязки дискретных цепей используются

- 1) разделительные трансформаторы;
- 2) оптические пары;
- 3) как разделительные трансформаторы, так и оптические пары.

17. Дискретные входы УСО контроллера в активном состоянии обычно потребляют ток порядка

- 1) 100 – 200 мкА;
- 2) 1 – 10 мА;
- 3) 100 – 200 мА.
- 4) Могут потреблять любой ток, так как его величина не имеет принципиального значения.

18. Сопротивление входов УСО по напряжению обычно имеет порядок

- 1) десятки Ом;
- 2) десятки КОм – несколько МОм;
- 3) сотни МОм.
- 4) Сопротивление может быть любым, так как его величина не имеет принципиального значения.

19. Сопротивление токовых входов УСО обычно

- 1) не превышает нескольких Ом;
- 2) не превышает одного-двух КОм;
- 3) не менее 100 КОм.

- 4) Сопротивление может быть любым, так как его величина не имеет принципиального значения.
20. Контроллер через дискретный выход управляет включением магнитного пускателя (катушка – переменного тока). Какой вариант цепи управления неправильный?
- 1) Семисторный выход – катушка пускателя.
  - 2) Транзисторный выход – промежуточное реле постоянного тока – катушка пускателя.
  - 3) Транзисторный выход – промежуточное реле переменного тока – катушка пускателя.
  - 4) Релейный выход – катушка пускателя.
21. Преимуществом транзисторного выхода перед релейным является
- 1) большее максимальное коммутируемое напряжение;
  - 2) больший максимальный коммутируемый ток;
  - 3) большая максимальная частота коммутации.
22. Преимуществом релейного выхода УСО перед транзисторным является
- 1) более высокая допустимая частота коммутации;
  - 2) возможность организации ШИМ выходного сигнала;
  - 3) возможность коммутировать цепи переменного тока.
23. Выходной сигнал измерительного преобразователя – 4-20мА. Вход УСО рассчитан на работу с сигналом 0-10 В и имеет большое (несколько МОм) входное сопротивление. Как согласовать сигналы?
- 1) Соединить ИП и УСО через последовательно включенное сопротивление 500 Ом, программно установить смещение (-20%);
  - 2) Соединить ИП и УСО, включив параллельно входу УСО сопротивление 500 Ом, программно установить смещение (-20%);
  - 3) Соединить ИП и УСО через последовательно включенное сопротивление 500 Ом, программно установить смещение (+20%);
  - 4) Соединить ИП и УСО, включив параллельно входу УСО сопротивление 500 Ом, программно установить смещение (+20%).
24. Контроллер имеет 32 дискретных входа. Сколько байт будет занимать в памяти контроллера информация об их состоянии?
- 1) 4;
  - 2) 16;
  - 3) 32;
  - 4) 64.
25. Точковый выход измерительного преобразователя ИП рассчитан на нагрузку сопротивлением не более 1000 Ом. Как подключить к преобразователю модуль УСО контроллера с входным сопротивлением не более 500 Ом и регистрирующий прибор РП с входным сопротивлением не более 300 Ом?



- 1) Подключить входы УСО и РП к выходу ИП параллельно для уменьшения суммарного сопротивления повышения точности измерений;
  - 2) Подключить входы УСО и РП к выходу ИП последовательно;
  - 3) Никакое соединение не обеспечит нормальной передачи сигнала.
26. Токовый выход измерительного преобразователя ИП рассчитан на нагрузку сопротивлением не более 1000 Ом. Как подключить к преобразователю модуль УСО контроллера с входным сопротивлением не более 500 Ом и регистрирующий прибор РП с входным сопротивлением не более 1000 Ом?
- 1) Подключить входы УСО и РП к выходу ИП параллельно;
  - 2) Подключить входы УСО и РП к выходу ИП последовательно;
  - 3) Никакое соединение не обеспечит нормальной передачи сигнала.
27. В IP-коде, обозначающем степень защиты корпуса, первая цифра обозначает
- 1) степень защиты от воды;
  - 2) степень защиты от твердых тел;
  - 3) степень защиты от внешнего теплового излучения;
  - 4) степень защиты от электромагнитного излучения.
28. В IP-коде, обозначающем степень защиты корпуса, вторая цифра обозначает
- 1) степень защиты от воды;
  - 2) степень защиты от твердых тел;
  - 3) степень защиты от внешнего теплового излучения;
  - 4) степень защиты от электромагнитного излучения.
29. Преимуществом РС-совместимых промышленных контроллеров и компьютеров является
- 1) их высокая надежность;
  - 2) их высокая функциональность;
  - 3) их «открытость»;
  - 4) все выше перечисленные преимущества.
30. Форм-фактор одноплатного компьютера не определяет
- 1) тип используемого микропроцессора;
  - 2) тип шины;
  - 3) размеры платы;
  - 4) тип используемых для подключения платы разъемов.
31. Форм-фактор одноплатного компьютера определяет
- 1) тип используемого микропроцессора;
  - 2) тип шины;
  - 3) типы и количество сетевых интерфейсов;

## **Раздел 2. Программные средства АСУ ТП**

1. Программирование промышленных контроллеров производится с помощью

- 1) SoftLogic-систем;
- 2) SCADA - систем;
- 3) DCS;
- 4) MES.

2. Программу контроллера, решающего задачу автоматического регулирования непрерывно изменяющейся величины, удобнее составить на языке

- 1) ST (Structured Text);
- 2) FBD (Function Block Diagram);
- 3) IL (Instruction List);
- 4) LD (Ladder Diagram).

3. Сложную релейно-контакторную систему управления решили заменить на современную систему на базе программируемого логического контроллера. На каком языке программирования удобнее и быстрее составить программу для контроллера?

- 1) ST (Structured Text).
- 2) FBD (Function Block Diagram).
- 3) IL (Instruction List).
- 4) LD (Ladder Diagram).

4. Какой из языков программирования контроллеров наиболее близок к языкам программирования высокого уровня, типа C, Pascal и т.д.?

- 1) ST (Structured Text);
- 2) FBD (Function Block Diagram);
- 3) IL (Instruction List);
- 4) LD (Ladder Diagram).

5. Какой из языков программирования контроллеров наиболее близок к языку Assembler или практически является таковым?

- 1) ST (Structured Text);
- 2) FBD (Function Block Diagram);
- 3) IL (Instruction List);
- 4) LD (Ladder Diagram).

6. Сторожевой таймер, применяемый в промышленных контроллерах и компьютерах это аппаратно-программное средство,

- 1) перехватывающее все прерывания от внешних устройств;
- 2) перезапускающее контроллер (компьютер) в случае зависания программы;
- 3) осуществляющее антивирусную защиту.

7. С какой целью в алгоритмах цифрового ПИД-регулирования используется балластное аperiodическое звено?

- 1) Для фильтрации высокочастотных шумов;

- 2) Для упрощения реализации закона регулирования (исключения процедуры вычисления производной);
- 3) Для решения обеих перечисленных выше задач, которые, по сути, представляют собой одну задачу;
- 4) Балластное апериодическое звено в алгоритмах цифрового ПИД-регулирования применять категорически нельзя.

8. При обработке сигналов расходомеров определенного типа в программах контроллеров используются алгоритмы

- 1) извлечения квадратного корня;
- 2) возведения в квадрат;
- 3) определения натурального логарифма.

9. Проблема безударности переключений возникает при разработке алгоритмов

- 1) программно-логического управления;
- 2) автоматического регулирования;
- 3) защиты и сигнализации.

10. Процедура «обратного счета», реализуемая в контроллерах для решения проблемы безударности переключений предусматривает

- 1) пересчет входных сигналов алгоритмических блоков по предполагаемым значениям их выходных сигналов;
- 2) пересчет предыдущих значений выходных сигналов алгоритмических блоков по их текущим значениям;
- 3) пересчет времени изменения выходных сигналов алгоритмических блоков по предыдущим значениям их входных сигналов.

11. НМІ (ММІ) это

- 1) средства отображения и представления технологической информации;
- 2) средства автоматического управления;
- 3) средства планирования производственного процесса.

12. Основное назначения SCADA-систем –

- 1) сбор данных, визуализация технологического процесса, супервизорное управление;
- 2) разработка, отладка и загрузка программ для промышленных контроллеров;
- 3) разработка проекта автоматизации технологического процесса.

13. Является ли SCADA-система системами класса ММІ (НМІ)?

- 1) Безусловно является;
- 2) Безусловно не является;
- 3) Является в зависимости от набора функций, реализованных в SCADA-системе.

14. Программное обеспечение, реализующее стандарт OPC (OLE for Process Control) используется в основном в

- 1) промышленных контроллерах;
- 2) SCADA-системах;
- 3) офисных приложениях административного уровня управления производством.

15. Система TRACE MODE позволяет программировать

- 1) любые промышленные контроллеры и компьютеры;
- 2) промышленные контроллеры и компьютеры любого типа, но только из списка поддерживаемого оборудования;
- 3) только PC-совместимые промышленные контроллеры и компьютеры.

16. SCADA-системой не является система

- 1) Genesis32;
- 2) TRACE MODE;
- 3) Ultralogic.

17. Программные системы управления основными фондами, техническим обслуживанием и ремонтами является системы класса

- 1) EAM (Enterprise Asset Management);
- 2) HRM (Human Resources Management);
- 3) MES (Manufacturing execution system).

18. Программные системы управления персоналом является системы класса

- 1) EAM (Enterprise Asset Management);
- 2) HRM (Human Resources Management);
- 3) MES (Manufacturing execution system).

### **Раздел 3. Промышленные компьютерные сети**

1. Недостатком всех централизованных автоматизированных систем управления технологическими процессами является

- 1) большой расход соединительных кабелей;
- 2) низкое быстродействие (большое время отклика);
- 3) низкая надежность.

2. Интеллектуальными (smart) устройствами в составе АСУТП называют

- 1) программируемые логические контроллеры, выполняющие функции контроля и регулирования технологических параметров;
- 2) операторские рабочие станции под управлением SCADA-систем;
- 3) измерительные преобразователи и исполнительные механизмы с микропроцессорными системами обработки данных и управления и промышленными сетевыми интерфейсами.

3. В модели компьютерных сетей ISO OSI сеть представлена уровнями, число которых

- 1) 5;

- 2) 7;
- 3) 8;
- 4) 9.

4. Какой из протоколов не является протоколом прикладного уровня компьютерной сети?

- 1) FTP;
- 2) HTTP;
- 3) Telnet;
- 4) TCP.

5. В URL «[www.amursu.ru](http://www.amursu.ru)» «www» – это

- 1) название сетевой службы;
- 2) протокол прикладного уровня;
- 3) имя узла.

6. Разбиением данных на пакеты при отправлении, сборкой пакетов с контролем последовательности их получения занимается

- 1) уровень приложений;
- 2) транспортный уровень (например, протокол TCP);
- 3) сетевой уровень (например, протокол IP);
- 4) канальный уровень.

7. Вопросами логической адресации узлов в глобальных сетях занимается программное обеспечение

- 1) уровня приложений;
- 2) транспортного уровня;
- 3) сетевого уровня;
- 4) канального уровня.

8. Вопросами физической адресации узлов в однородных локальных сетях занимается программное и аппаратное обеспечение

- 1) уровня приложений;
- 2) транспортного уровня;
- 3) сетевого уровня;
- 4) канального уровня;
- 5) физического уровня.

9. В Ethernet и Fast Ethernet используется метод доступа к каналу связи

1) CSMA/CD – Carrier Sense Multiple Access with Collision Detection (множественный доступ с контролем несущей и обнаружением конфликтов);

2) TPMA – Token Passing Multiple Access (множественный доступ с передачей маркера);

3) TDMA – Time Division Multiple Access (множественный доступ с разделением во времени).

10. Переключатель (switch) – это устройство, работающее на уровнях

- 1) приложений;
- 2) транспортном и сетевом;
- 3) сетевом и канальном;
- 4) канальном и физическом.

11. Fieldbus – это

- 1) протокол передачи данных;
- 2) сетевая технология;
- 3) сфера применения сетевых технологий (промышленное производство).

12. Какие из требований к промышленным сетям противоречат друг другу?

- 1) помехоустойчивость и возможность получения «распределенного интеллекта» путем предоставления максимального доступа к каналу нескольким ведущим узлам;
- 2) производительность и предсказуемость времени доставки информации;
- 3) минимальная стоимость устройств аппаратной реализации и доступность и простота организации физического канала передачи данных.

13. Какие уровни сетевой модели ISO OSI в большинстве случаев не используются в промышленных сетях?

- 1) прикладной, представления, сеансовый;
- 2) представления, транспортный, сетевой;
- 3) сетевой, канальный, физический.

14. Программные и аппаратные средства промышленных сетей в большинстве случаев ограничиваются следующими уровнями сетевой модели ISO OSI

- 1) прикладной, транспортный, сетевой, физический;
- 2) прикладной, сетевой, канальный, физический;
- 3) прикладной, канальный, физический.

15. Какой из видов кабеля наиболее часто применяется в промышленных сетях?

- 1) Коаксиальный кабель;
- 2) Витая пара;
- 3) Оптоволокно.

16. Какой из видов кабеля обеспечивает максимальную защиту от электромагнитных помех?

- 1) Коаксиальный кабель;
- 2) Витая пара;
- 3) Оптоволокно.

17. Какой из стандартных физических интерфейсов наиболее часто применяется в промышленных сетях?

- 1) RS-232;
- 2) RS-422;
- 3) RS-485.

18. Промышленная сеть связывает множество датчиков, выходная информация которых имеет дискретный характер. Какой из типов сообщений обеспечит максимальную производительность сети?

- 1) Опрос;
- 2) Широковещательное стробирование;
- 3) Периодическая рассылка;
- 4) Отсылка сообщений по факту изменения состояния.

19. В сети, построенной по принципу MASTER-SLAVE, только MASTER-устройства могут

- 1) передавать сообщения;
- 2) инициализировать обмен сообщениями;
- 3) допускать настройку параметров обмена сообщениями.

20. Какая из промышленных сетевых технологий позволяет создавать мультимастерные сети?

- 1) ModBus;
- 2) CAN;
- 3) ASI.

21. В какой из промышленных сетевых технологий сообщения имеют минимальный размер?

- 1) ModBus;
- 2) CAN;
- 3) ASI;
- 4) ProfiBus.

22. Сколько MASTER-устройств может присутствовать в сети, построенной по протоколу HART?

- 1) один;
- 2) не более двух;
- 3) до 32;
- 4) неограниченное количество.

23. В какой из промышленных сетевых технологий «несущим» сигналом является сигнал 4-20мА?

- 1) CAN;
- 2) ASI;
- 3) HART;
- 4) ProfiBus.

24. В какой из промышленных сетевых технологий поддерживается передача данных на расстояния выше 3 км?

- 1) CAN;
- 2) ASI;
- 3) HART;
- 4) ProfiBus.

25. С точки зрения применения в условиях промышленности к недостаткам технологии Ethernet следует отнести

- 1) большие издержки при передаче данных небольшого объема;
- 2) недостаточная дальность передачи;
- 3) недостаточная скорость передачи.

26. В основе сетевой промышленной шины DeviceNet лежит стандарт

- 1) CAN;
- 2) ASI;
- 3) HART;
- 4) ProfiBus.

27. Какая из перечисленных технологий обеспечивает максимальную скорость передачи данных?

- 1) CAN;
- 2) ASI;
- 3) ProfiBus;
- 4) Традиционная технология 4–20 мА.



## **10. ВОПРОСЫ К ЭКЗАМЕНУ И ЗАЧЕТУ**

Контрольные вопросы к экзамену и зачёту представлены в п.6 рабочей программы ИСПУ.

## **11. КАРТА ОБЕСПЕЧЕННОСТИ ДИСЦИПЛИНЫ КАДРАМИ ПРОФЕССОРСКО-ПРЕПОДАВАТЕЛЬСКОГО СОСТАВА.**

Лекции и лабораторные работы ведет ассистент кафедры «Автоматизация производственных процессов и электротехники» -  
Русинов Владислав Леонидович.