ВВЕДЕНИЕ В СКАДА СИСТЕМЫ

HTTP://BNBARS.MOY.SU

ГЛАВА 1. ГРАФИЧЕСКИЙ ИНТЕРФЕЙС

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.

Но, тем не менее, каждая SCADA - система по-своему уникальна и, несмотря на поддержание стандартных функций, обладает присущими только ей особенностями. При рассмотрении графических возможностей SCADA - систем InTouch и Citect предполагается обратить внимание не только на возможности инструментариев по созданию графических объектов, но и на другие предоставляемые пользователю услуги, облегчающие и ускоряющие процесс разработки приложений (проектов).

1.1. Графические средства InTouch

Компоненты среды разработки InTouch:

- WindowMaker - инструментальная среда разработки приложений;
- Application Explorer - представление приложения в иерархическом виде с доступом к любому компоненту приложения и многим часто используемым командам и функциям WindowMaker.

Проект, созданный в пакете InTouch, представляет собой набор окон (Window) с различными графическими и текстовыми объектами.

1.1.1. Окна

Свойства каждого окна (наличие заголовка, цвет фона, размеры и т. д.) определяются при его создании. Создание нового окна производится в среде разработки WindowMaker щелчком по

иконке панели инструментов General или командой File/New Window. На экране появится диалог Window Properties (Свойства окна, рис. 1.1.1).

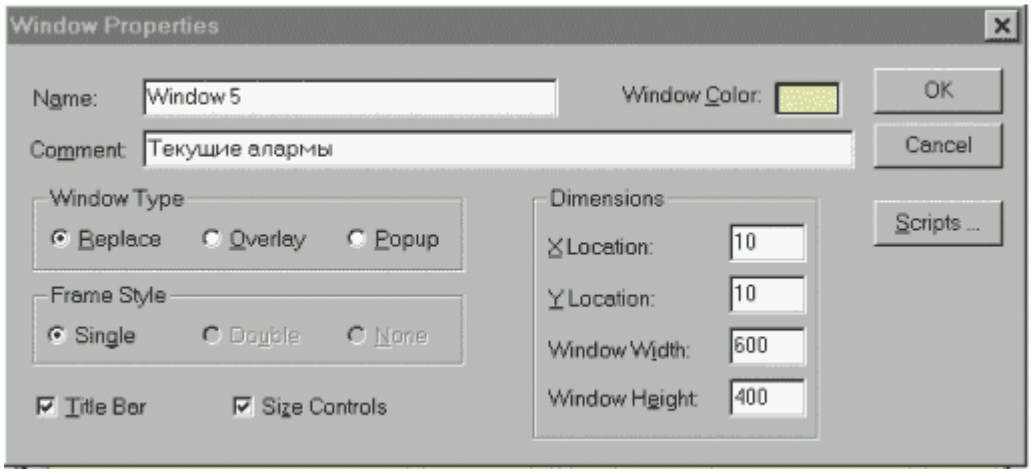


Рис. 1.1.1. Диалог Window Properties (Свойства окна).

Каждое окно должно иметь свое имя для его идентификации в приложении (Name). Цвет фона создаваемого окна выбирается из цветовой палитры, вызываемой на экран щелчком по окошку Window Color. В поле Comment можно ввести комментарий, связанный с этим окном (необязательно). Эта информация нужна только для документирования и не используется приложением. InTouch предлагает три типа окон (Window Type):

- Replace (заменяющее) - закрывает все существующие окна, перекрываемые им при появлении на экране, включая окна типа Popup и другие окна типа Replace.
- Overlay (перекрывающее) - появляется поверх всех отображаемых в текущий момент окон. Когда окно типа Overlay закрывается, все скрываемые им окна восстанавливаются. Щелчок мыши по любому видимому участку лежащего ниже окна приводит к переходу его на передний план.
- Popup (всплывающее) - похоже на окно типа Overlay, только оно всегда остается поверх всех других открытых окон. Окно закрывается после соответствующей команды пользователя.

Выбор типа создаваемого окна производится включением соответствующей кнопки в поле Window Type. В поле Frame Style (стиль обрамления) выбирается необходимый стиль обрамления окна:

- Single - окно с рамкой, допускается заголовок;
- Double - окно с рамкой без заголовка;
- None - окно без рамки и заголовка.

Чтобы у окна была полоса с заголовком, где выводится имя окна, включают опцию Title Bar. Эта полоса также служит для перемещения окна при захвате ее мышью. При выборе этой опции отключатся опции Double и None для стиля обрамления. Для возможности изменения размеров окна, когда оно откроется в WindowMaker, следует выбрать опцию Size Controls (управление размером). В группе полей Dimentions определяются текущие размеры и положение окна на рабочем поле:

- X Location - расстояние в пикселях между левым краем рабочего поля WindowMaker и левым краем описываемого окна;
- Y Location - расстояние в пикселях между верхним краем рабочего поля WindowMaker и верхним краем описываемого окна;
- Window Width - ширина окна в пикселях;
- Window Height - высота окна в пикселях.

По умолчанию при создании нового окна эти параметры примут значения предыдущего (последнего) созданного окна. Кнопка Scripts (скрипты) дает возможность войти в диалог Window Script для создания оконного сценария. Для унификации внешнего вида окон приложения и сокращения сроков разработки приложений InTouch предлагает несколько приемов. Один из таких приемов - дублирование окон. Создание копий окон выполняется командой File/ Save Window As. Для быстрого доступа к этой команде можно воспользоваться меню правой кнопки мыши (см. ниже). Второй прием, который также позволяет экономить время разработки приложения - импорт окон. Можно повторно использовать все ранее созданные окна, объекты и скрипты. Чтобы импортировать окна из другого InTouch - приложения, необходимо воспользоваться командой File/ Import. Интерфейс WindowMaker с открытым окном представлен на рис. 1.1.2.

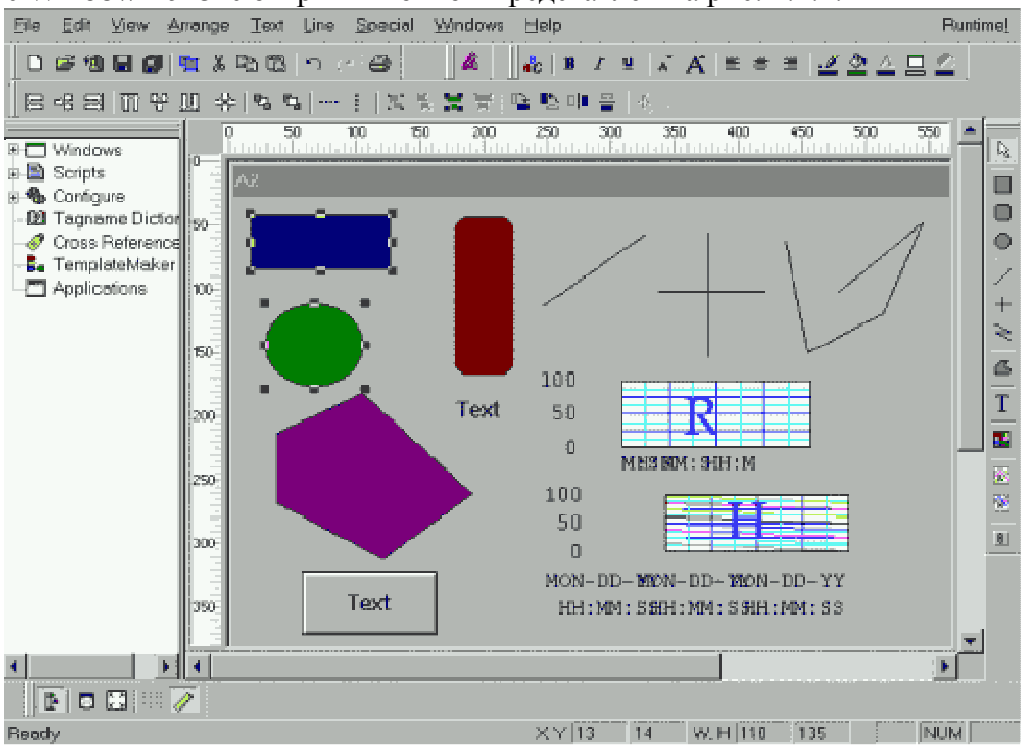


Рис. 1.1.2. Интерфейс WindowMaker.

Сверху экрана расположена строка меню, включающая опции для работы с окнами, редактирования и выравнивания объектов в окне, настройки инструментариев, текста, толщины и стиля линий и т. д. Слева от рабочего поля видно меню Application Explorer, которое может быть выведено в интерфейс WindowMaker или закрыто нажатием соответствующей иконки инструментария.

1.1.2. Инструментарий InTouch

Инструментарий InTouch представлен пятью инструментальными панелями, сгруппированными по функциональному принципу.

Панель General содержит элементы, соответствующие часто используемым командам меню File и Edit. Эти элементы известны читателю по работе в среде Windows и не требуют дополнительного пояснения.

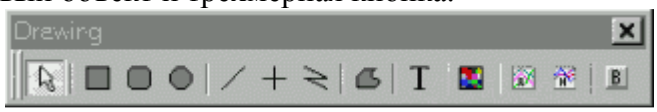
В панель форматов Format включены средства, выполняющие большую часть команд

форматирования текстовых объектов меню Text. Она содержит также средства выбора цвета линии, текста, заполнения объекта, фона окна и цвета "прозрачных объектов".

Панель выравнивания Arrange содержит инструменты, соответствующие командам выравнивания объектов меню Arrange.

В нее включены кнопки для выполнения команд выравнивания объектов, размещения на переднем и заднем плане, равномерной расстановки объектов по горизонтали и вертикали, объединения отдельных объектов в символы и компоненты и их разъединения, вращения по и против часовой стрелке на 900, зеркального отображения объектов по горизонтали и вертикали.

Панель рисования Drawing включает инструменты для создания простых и сложных объектов. Первые восемь инструментов и последний предназначены для создания простых объектов: прямоугольник (квадрат), скругленный прямоугольник, окружность (эллипс), прямая линия под любым углом, горизонтальная и вертикальная прямая, ломаная линия, многоугольник, текстовый объект и трехмерная кнопка.



С помощью остальных трех инструментов панели Drawing могут быть созданы сложные объекты операторских интерфейсов: контейнер для вставки растровых изображений, тренд реального времени и архивный тренд. На рис. 1.1.2 на рабочем поле окна WindowMaker показаны примеры объектов, созданных инструментами панели Drawing. В панели View (Вид) представлено всего пять кнопок (слева направо):

- кнопка, соответствующая команде отображения/закрытия окна Application Explorer;
- кнопка, соответствующая команде Hide All (спрятать все), относящейся к панелям инструментов;
- кнопка переключения обычного изображения окна в полноэкранное и обратно;
- кнопка, соответствующая команде Snap to Grid (привязать к координатной сетке);
- кнопка отображения/закрытия линейки окна WindowMaker (рис. 1.1.2).

Все инструментальные панели могут быть "закреплены" у любого края окна WindowMaker, в том числе и панель рисования. При необходимости их можно переместить внутрь рабочего поля.

При разработке операторских интерфейсов достаточно важно обеспечить пользователю удобный и быстрый доступ к наиболее часто используемым командам. В этом плане InTouch предлагает ряд меню, вызываемых на экран нажатием правой кнопки мыши. Эти меню могут относиться к окнам (на рис. 1.1.3 слева), графическим объектам (в середине) и полям диалогов (справа).

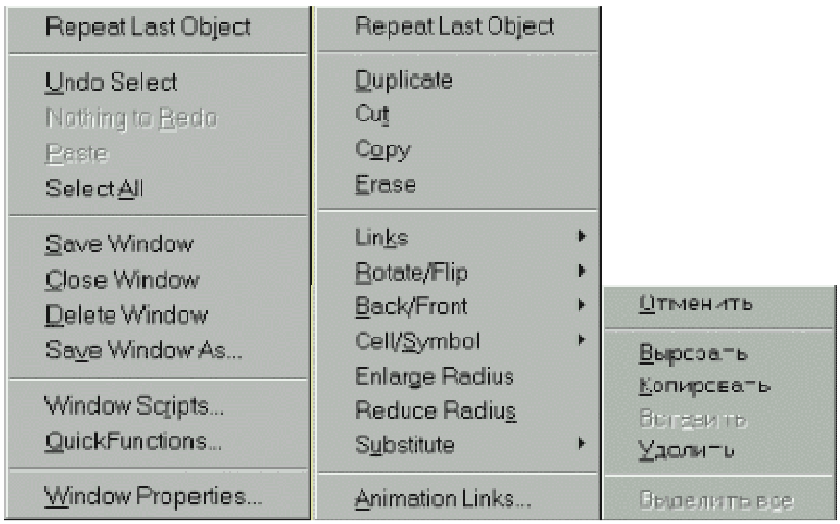


Рис.1.1.3. Меню для окна, объекта и диалога, активизируемые правой кнопкой мыши.

1.1.3. Объекты и их свойства

Простые объекты.

WindowMaker поддерживает четыре базовых типа простых объектов: линии, заполненные контуры, текст и кнопки. Каждый из этих простых объектов имеет свойства, влияющие на его внешний вид. Такими свойствами являются цвет линии, цвет заполнения, высота, ширина, ориентация и т. д., и они могут быть статическими или динамическими.

- Линия - это объект, представляющий собой один или несколько связанных отрезков. Толщина линии и ее стиль являются статическими свойствами линии, присваиваемыми ей во время создания, и лишь цвет линии может быть связан с анимационной функцией.
- Заполненный контур (прямоугольник, скругленный прямоугольник, круг, эллипс, многоугольник) представляет собой двухмерный объект. К динамическим свойствам такого объекта относятся цвет контурной линии, цвет заполнения, насыщенность цвета заполнения, высота, ширина, расположение, видимость и ориентация.
- Текст представляет собой последовательность символов. К статическим свойствам текста относятся тип шрифта, его размер, выделение, курсив, подчеркивание, выравнивание. Анимационные свойства шрифта - цвет, видимость и расположение.
- Кнопка - часто используемый объект при создании операторских интерфейсов. С кнопками могут быть связаны функции различных типов. Нажатие кнопки может вызвать выполнение скриптов, кнопкой можно производить ввод аналоговых и дискретных величин и т. д.

Текст на кнопке редактируется с помощью команды Special/Substitute Strings... При этом текстовое поле может содержать только одну строку.

Один и тот же объект может иметь набор различных динамических свойств. Комбинации этих свойств предоставляют возможность создавать на экране в режиме исполнения (Runtime) практически любые динамические эффекты. Для установки динамических свойств надо прежде всего вызвать на экран диалог их выбора (рис.1.1.4). Это достигается командой Special/Animation Link или двойным щелчком левой кнопки мыши на объекте.

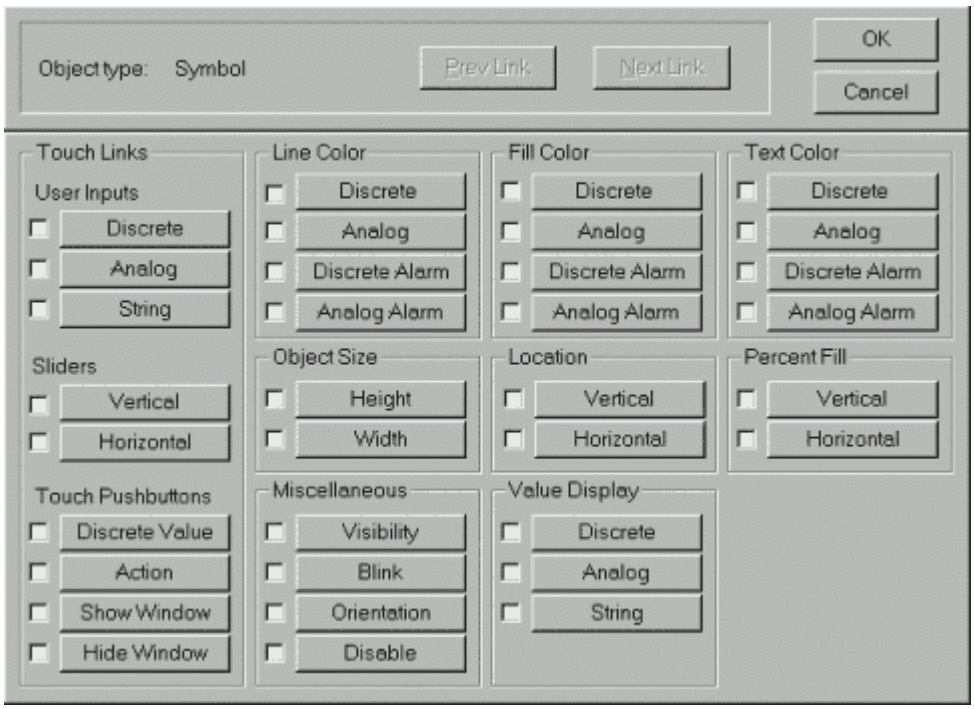


Рис.1.1.4. Диалог выбора динамических свойств объекта.

Все динамические связи можно разделить на две группы: Touch Links (левая колонка) и Display Links (три колонки справа). С помощью свойств Touch Links выполняется какой - либо ввод в систему. Свойства Display Links осуществляют вывод информации на экран дисплея.

Нажатие на любую клавишу диалога (рис.1.1.4) вызывает появление нового диалога для определения соответствующего свойства объекта. Количество диалогов соответствует количеству динамических свойств (кнопок) диалога выбора. Все диалоги различны, но большинство из них имеет общие характеристики:

- окно типа объекта;
- одинаковую палитру цветов;
- быстрый вызов словаря переменных;
- быстрый доступ к полям переменных;
- поддержку правой кнопки мыши в полях Tagname (имя переменной) и Expression (выражение).

На рис.1.1.5 приведен диалог для определения свойств объекта (кнопки), управляющего значением дискретной переменной.

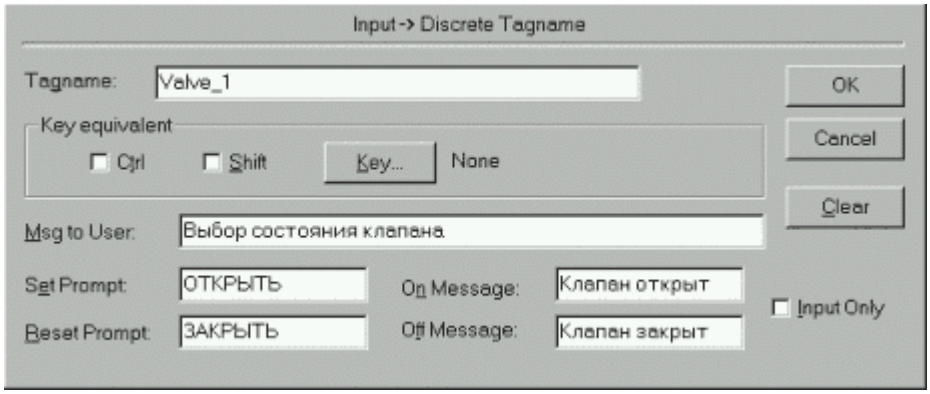


Рис.1.1.5. Диалог определения свойств кнопки.

Завершение работы с диалогом производится нажатием кнопки Ok. Если переменная поля Tagname была ранее определена в словаре переменных данного приложения,

пользователь возвращается в диалог выбора динамических свойств объекта (рис.1.1.4). Можно либо продолжить определение других динамических свойств для данного объекта, либо, нажав Ok, вернуться на поле разработки окна приложения.

Сложные объекты.

- Символ - это некоторая комбинация простых объектов, которые обрабатываются как один объект. Любое изменение статических или динамических свойств символа влияет на все составляющие символа. Например, если создать символ "насос" из двух кругов и двух прямоугольников и присвоить ему динамическое свойство Fill Color (цвет заполнения), то это свойство будет распространяться на все четыре простых объекта. Различные объекты символа могут иметь разные значения одного и того же свойства, если они были присвоены этим объектам до объединения в символ. Bitmap - объекты, кнопки, компоненты не могут быть включены в состав символа.
- Компонент - это совокупность двух или более объектов, символов или других компонентов, образующих единый элемент. Они создаются путем выбора двух и более объектов, символов или компонентов и последующего запуска команды Arrange/Make Cell. Компоненты реализуют пространственную взаимосвязь между составляющими их графическими элементами. Каждая составляющая компонента может иметь свои собственные динамические свойства. Компоненты используются для таких виртуальных устройств, как панель управления контроллером, движковый регулятор и т. д. Компонент не может менять свой размер, ему нельзя присваивать динамические свойства (внутри компонента есть объекты и символы со своими динамическими свойствами). Нельзя изменять и статические свойства (внешний вид). Для изменения статических и динамических свойств компонента его надо разобрать на составные части командой Arrange/Break Cell. Однако компоненты можно дублировать, копировать, вставлять, выравнивать, перемещать и т. д.
- Мастер-объект - это предварительно созданный компонент с определенными статическими и динамическими свойствами, находящийся в библиотеке мастер-объектов (Wizards) и доступный для многократного применения. Но, в отличие от компонента, динамические свойства которого настраиваются для каждой составляющей отдельно до объединения в компонент, динамические свойства мастер-объекта быстро настраиваются с помощью специализированного диалога. Другими словами, фирма Wonderware провела большую работу и создала огромное количество мастер-объектов (несколько тысяч), определив для каждого из них механизм быстрой настройки статических и динамических свойств. Все эти мастер-объекты разделены на большое количество групп и размещены в соответствующей библиотеке. Доступ к ней осуществляется нажатием иконки Wizard в интерфейсе WindowMaker, что вызывает появление на экране диалога Wizard Selection (Выбор мастер-объекта, рис. 1.1.6).

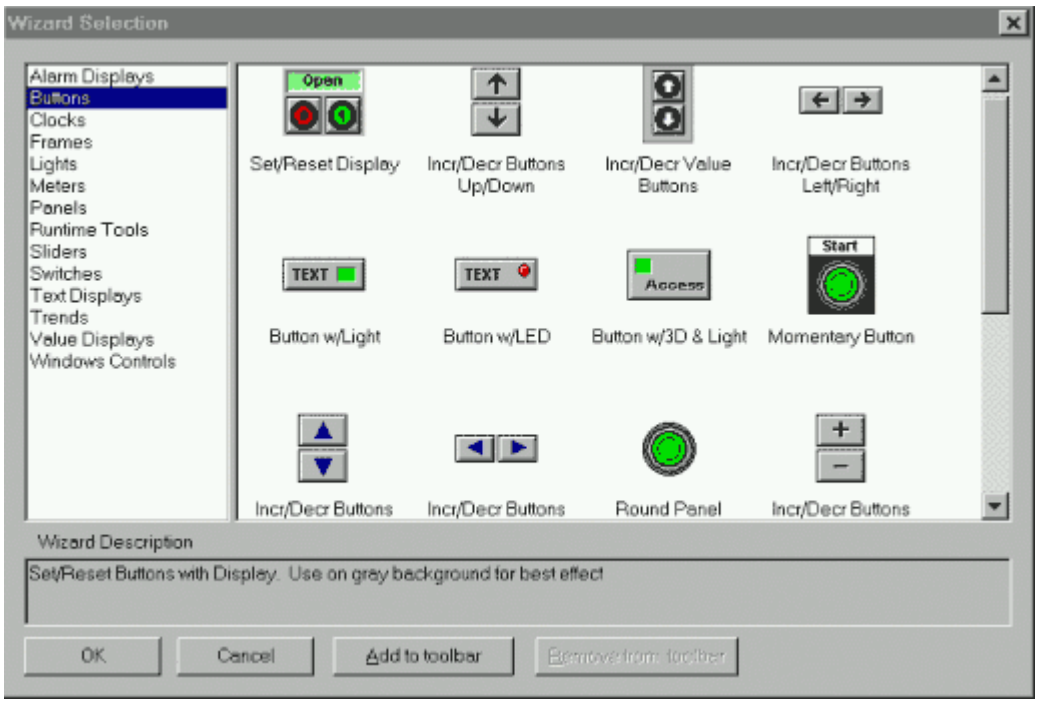
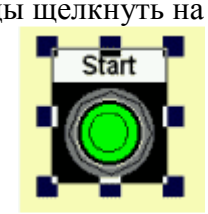


Рис.1.1.6. Диалог Wizard Selection (выбор мастер-объекта).

В левой части диалога - список групп мастер-объектов, включающий такие категории, как Buttons (кнопки), Sliders (ползунковые регуляторы), Switches (переключатели) и т. д.

В правой части диалога приведены все мастер-объекты выбранной в данный момент группы (на рис. 1.1.6 - кнопки). Двойной щелчок по требуемому мастер-объекту возвращает пользователя в окно разработки приложения. Курсор принимает форму уголка с символом. Наконец, щелчок мыши на свободном месте окна приводит к появлению мастер-объекта в окне приложения. Для его конфигурирования (определения динамических свойств) следует дважды щелкнуть на мастер-объекте.



Например, двойной щелчок по кнопке Momentary Button (кнопка запуска), предварительно вставленной в окно приложения, выводит на экран диалог конфигурирования этой кнопки (рис.1.1.7).

Достаточно ввести имя дискретной переменной, желаемый текст на кнопке, отметить несколько опций и нажать Ok.

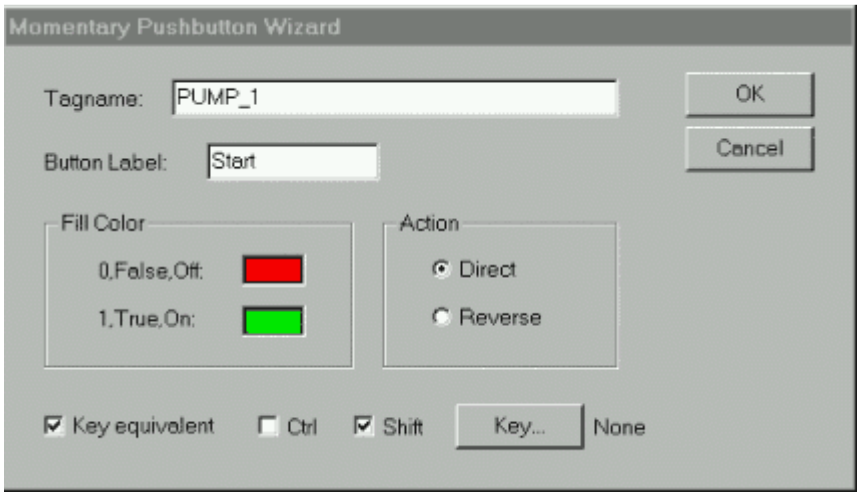


Рис.1.1.7. Диалог конфигурирования кнопки запуска.

Инструмент Bitmap инструментальной панели рисования позволяет копировать и встраивать в приложение InTouch растровые объекты (совокупность точек). С помощью него создается "контейнер" для последующей вставки объекта из папки обмена Windows либо файлов с расширением .BMP, .JPG, .PCX, .TGA. Для WindowMaker растровое изображение является единым объектом. Невозможно ни анимировать его отдельные части, ни вставлять Bitmap - объекты в символы (можно вставлять в компоненты). Такой объект можно развернуть на рабочем поле на 90, 180, 270, 360 градусов, а также определить для него цвет "прозрачности", чтобы через него можно было видеть и другие объекты.

Тренды. InTouch предлагает пользователю два сложных объекта типа тренд: тренд реального времени и исторический (архивный) тренд. Эти объекты позволяют отображать в виде графиков значения данных реального времени (4 пера) и архивных данных (8 перьев). Оба типа трендов создаются при использовании специальных инструментов панели рисования окна WindowMaker с последующим конфигурированием. Подробная информация по созданию и конфигурированию трендов будет приведена в соответствующей главе.

1.2. Графические средства Citect

Компоненты среды разработки Citect:

- Citect Explorer - представление списка проектов и их стандартных папок в иерархическом виде с доступом к любому компоненту проекта;
- Project Editor (редактор проектов) - среда создания, конфигурирования и редактирования задач, не связанных с графическими страницами проекта;
- Graphics Builder (построитель интерфейсов) - среда создания и редактирование графического интерфейса;
- Cicode Editor (редактор Cicode) - полнофункциональная интегрированная среда для создания и отладки программ на языке Cicode.

Проект Citect обычно состоит и целого ряда страниц (Pages), которые выводятся на экран компьютера. Эти графические страницы обеспечивают "окно в процесс". С помощью графических страниц происходит процесс взаимодействия оператора с системой управления, в том числе восприятие данных и ввод управляющих воздействий. Важно создать графические страницы таким образом, чтобы они охватывали весь технологический процесс и предоставляли оператору всю необходимую для управления информацию. Причем процесс создания графических страниц проекта должен быть максимально упрощен, и разработчика надо снабдить полным и удобным инструментарием.

Citect предлагает разработчику следующие возможности:

- шаблоны большинства типов наиболее часто используемых страниц (окон);
- инструментарий для создания и динамизации графических объектов;
- специальный редактор - Bitmap Editor для создания точечных изображений;
- библиотеку статических объектов (Library Objects);
- библиотеку джинов и суперджинов.

1.2.1. Шаблоны окон операторского интерфейса

Ниже приведено описание некоторых шаблонов Citect, хранящихся в библиотеке:

- Blank - шаблон пустой страницы;
- Normal - шаблон базовой страницы для создания мнемосхем технологических процессов;
- PageMenu - шаблон для создания страницы меню, которая позволяет оператору быстро переходить к другим страницам или группам страниц проекта;
- BookMenu - шаблоны для создания меню в формате книг;

- TabMenu - шаблоны для создания меню в формате таблиц;
- Single Trend - шаблон для создания страницы с одним окном трендов, в котором имеется до 8 перьев;
- Double Trend - шаблон для создания страницы с двумя окнами трендов, в каждом из которых имеется до 8 перьев;
- Compare Trend - шаблон для создания страницы с двумя трендами, наложенными один на другой в целях их сравнения;
- Pop Trend - шаблон для создания маленькой страницы трендов, которая будет играть роль выпадающей страницы;
- Alarm - шаблон для создания страницы текущих алармов;
- Summary - шаблон для создания страницы сводки алармов;
- Hardware - шаблон для создания страницы аппаратных алармов.

Некоторые шаблоны страниц (для вывода алармов, трендов и статистических графиков) уже сконфигурированы и остается лишь ввести имена требуемых параметров. Независимо от выбранного шаблона в нем уже представлены все необходимые элементы: рамки, линейки и т. д.

Последовательность расположения страниц в проекте определяется при проектировании системы управления в диалоге Properties (Свойства страницы). С помощью средств навигации (клавиш) оператор имеет возможность последовательно переходить с одной страницы на другую в порядке возрастания (клавиша Next) или убывания (Prev). Всегда под рукой у оператора находятся клавиши перехода на страницы алармов (текущие алармы, аппаратные алармы и сводка алармов).

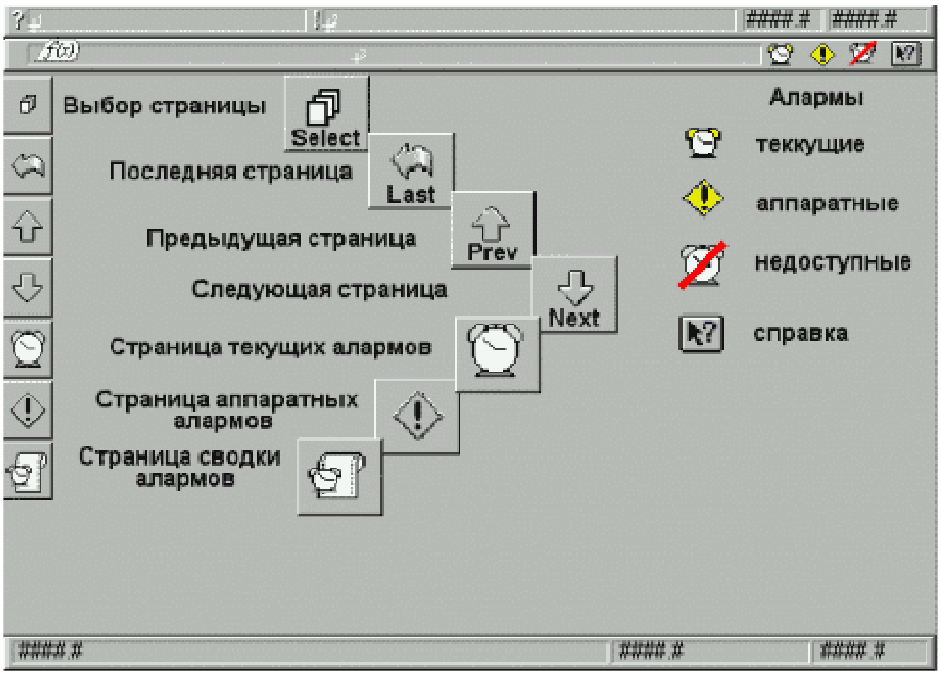


Рис. 1.2.1. Шаблон страницы проекта.

Для быстрого перехода на произвольную страницу предусмотрена клавиша Select (выбор). В каждом шаблоне страницы представлены средства отображения аварийных ситуаций и кнопка вызова справочной системы (рис.1.2.1).

Доступ к диалогу для выбора типа шаблона осуществляется из Citect Explorer (Project Editor) выбором соответствующей папки (команды).

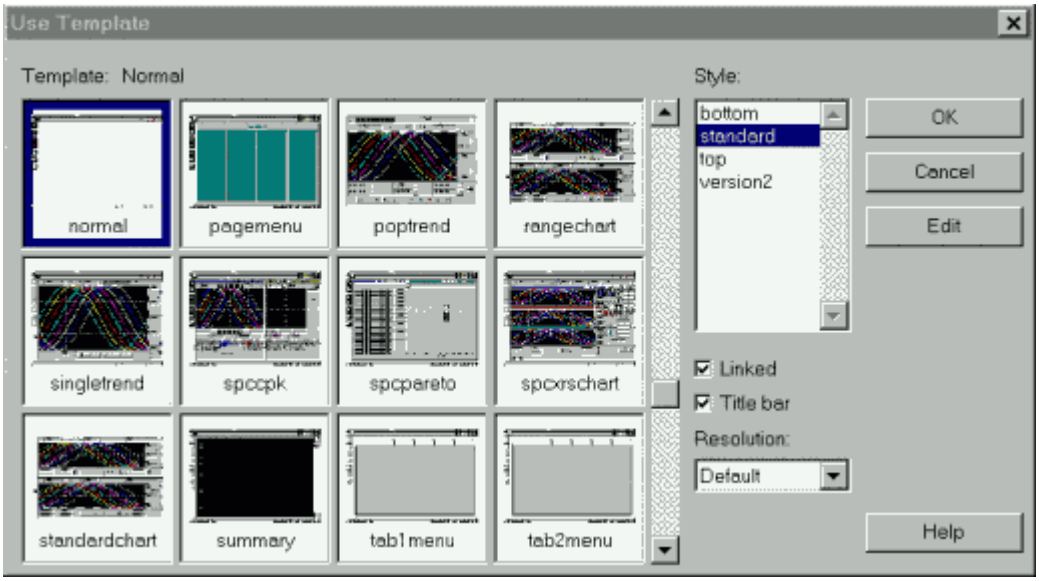


Рис.1.2.2. Диалог выбора шаблона.

По умолчанию в открывшемся диалоге (рис.1.2.2) выбран шаблон типа Normal, позволяющий создавать уникальные окна. Выше было сказано, что в шаблоне этого типа разрабатываются мнемосхемы технологических процессов.

Включение опции Linked в правом нижнем углу шаблона означает установку связи между будущей страницей проекта и шаблоном. При всех изменениях, внесенных в шаблон, она будет теперь автоматически обновляться.

Готовые шаблоны страниц (окон) - это безусловный плюс SCADA - пакета, так как они позволяют разработчику экономить значительное время. Если несколько страниц проекта созданы на базе одного и того же шаблона, легко модифицировать сразу всю эту группу страниц, производя изменения только в шаблоне. При включенной опции Linked все страницы группы изменятся автоматически. Наконец, применение в проекте типовых шаблонов позволяет унифицировать внешний вид страниц проекта, что положительно скажется на работе оператора. Щелчок по клавише Ok диалога выбора шаблонов переносит читателя в построитель интерфейсов Graphics Builder.

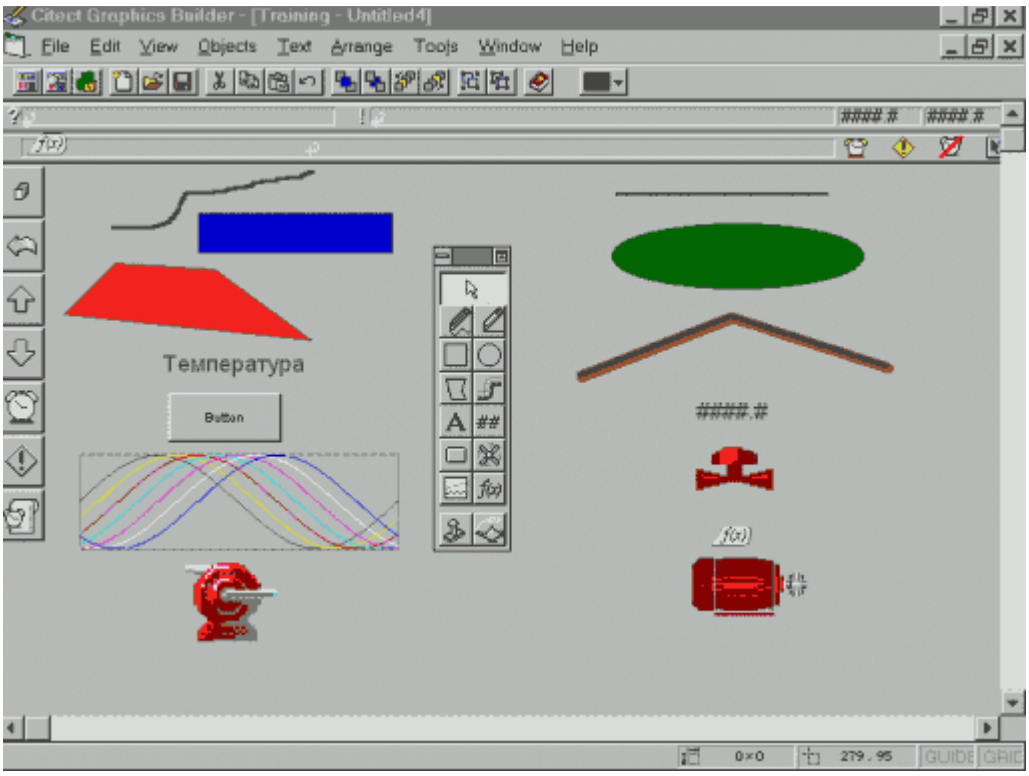


Рис.1.2.3. Страница на базе шаблона Normal с примерами объектов.

На рабочем поле окна Graphics Builder (рис.1.2.3) размещен шаблон Normal с инструментарием. Слева и справа от инструментария представлены примеры объектов, созданных с помощью различных инструментов.

1.2.2. Инструментарий

Инструментарий включает 14 различных инструментов и селектор (стрелка) для выбора объектов в окне.

Действия, необходимые для рисования объектов с помощью инструментов, представленных выше, очень просты и могут быть быстро освоены.

Для рисования таких объектов, как прямоугольник (квадрат), окружность (эллипс), кнопка, тренд, надо щелкнуть левой кнопкой мыши по соответствующему инструменту, подвести курсор к выбранному месту рабочего поля и, нажав и удерживая левую кнопку мыши, растянуть объект до требуемых размеров.

Выбор инструментов "вставка объекта" или "вставка джина" открывает соответствующую библиотеку. Следует выбрать объект для вставки на графическую страницу и щелкнуть Ok.

При вставке объектов с заданием динамических свойств после выбора этого инструмента предлагается сначала щелкнуть по рабочему полю, что вызовет на экран диалог для конфигурирования свойств объекта. Из этого диалога имеется доступ в библиотеку статических объектов.

После размещения объекта, созданного любым из инструментов, на странице автоматически появляется соответствующий диалог настройки свойств объекта. Объекты, созданные такими инструментами, как кнопка, тренд, вывод данных, вставка символов с заданием динамических свойств, выражение Cicode, вставка джинна, требуют настройки свойств.

Для редактирования свойств объектов следует дважды щелкнуть левой кнопкой мыши на соответствующем объекте.

Над объектами можно проделывать операции в Graphics Builder, используя меню Edit (правка), View (вид), Text (текст) и Arrange (выравнивание). Как и в других графических пакетах, объекты можно вращать, масштабировать, группировать, выравнивать и т. д.

В системе Citect набор свойств для большинства объектов - стандартный:

- перемещение (Movement) - горизонтальное, вертикальное, вращательное;
- размер (Size) - горизонтальный, вертикальный;
- цвет заполнения (Colour Level Fill) и изменение цвета (Colour change);
- команды по нажатию (Touch);
- команды клавиатуры (Keyboard Commands);
- ползунковый регулятор (Slider) - горизонтальный, вертикальный, вращательный;
- видимость (Visibility);
- блокировка (Disable);
- управление доступом (Access control).

Свойства объектов определяются в диалоге Свойства (рис. 1.2.4.).

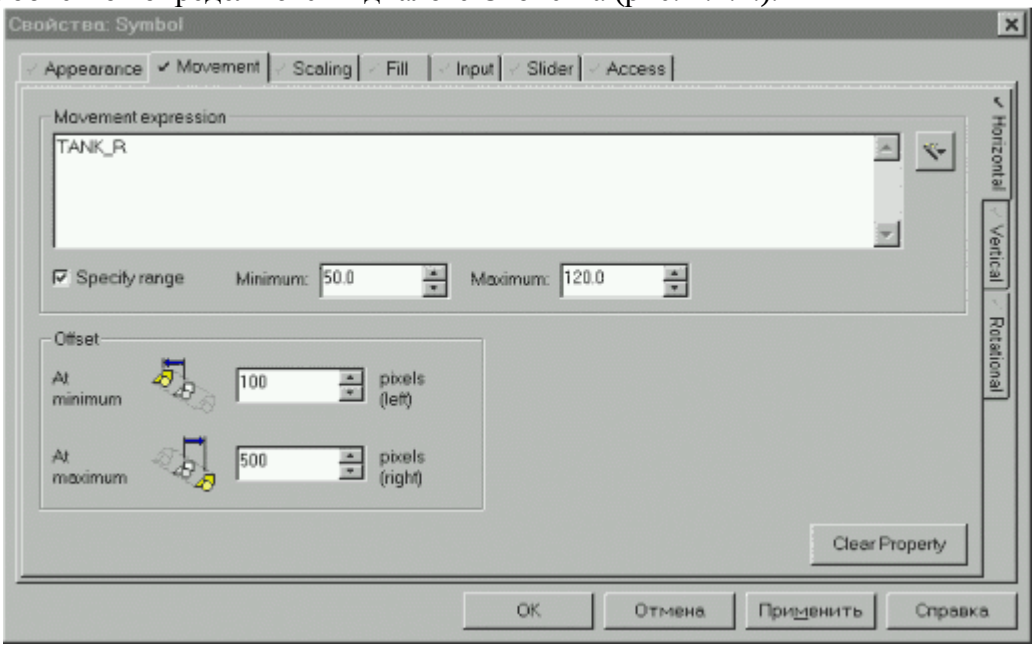


Рис.1.2.4. Диалог Свойства объекта с открытой закладкой Movement.

Таким образом, разработчик имеет дело с одними и теми же диалогами при конфигурировании свойств графических объектов. Это дает ряд преимуществ:

- скорость обучения новых пользователей существенно возрастает;
- упрощается работа по созданию графического интерфейса;

- сокращаются сроки разработки проекта.

Диалог Свойства объекта содержит несколько закладок (рис. 1.2.4): Appearance (вид), Movement (перемещение), Scaling (масштабное изменение размеров), Fill (заполнение), Input (ввод), Slider (ползунок), Access (доступ).

Нажатие любой из этих закладок выводит на экран соответствующий диалог для конфигурирования свойств объекта. Закладка Appearance определяет характеристики внешнего вида объекта: тип контурной линии (толщина линии, тип, цвет), цвет заполнения, тень и т. д. Здесь же определяется видимость объекта для оператора (объект может появиться на экране или исчезнуть в зависимости от выполнения некоторого условия).

Объекты или группы объектов могут перемещаться в режиме исполнения (Runtime) при изменении значения переменной или выражения. По умолчанию при увеличении значения этого выражения объект перемещается вправо, а при уменьшении значения - влево.

В диалоге Movement/Horizontal (горизонтальное перемещение, рис.1.2.4) предлагается определить переменную или выражение, вызывающее перемещение объекта (поле Movement Expression), его минимальное и максимальное значения (Minimum, Maximum), а также расстояния в пикселях, на которые будет перемещаться объект влево (Left) при принятии выражением минимального значения и вправо (Right) при принятии выражением максимального значения (поле Offset).

Ширина объекта или группы объектов может динамически изменяться в режиме исполнения при изменении значения некоторого выражения. При увеличении/уменьшении значения выражения ширина объекта увеличивается/уменьшается, соответственно. В диалоге Scaling предлагается определить выражение, вызывающее изменение ширины объекта, его минимальное и максимальное значения, а также минимум и максимум ширины объекта в процентах от ширины нарисованного объекта.

На закладке Fill определяются степень (уровень) заполнения объекта или его цвет в зависимости от значения выражения или переменной в режиме исполнения. На рис.1.2.5 приведен диалог Fill/Level (уровень заполнения), где предлагается определить переменную или выражение, определяющее изменение уровня в объекте (поле Level Expression), минимальное и максимальное значения (Minimum, Maximum) уровня, процент закрашивания объекта при минимальном и максимальном значениях уровня (поле Level), а также направление закрашивания объекта (кнопки Fill Direction).

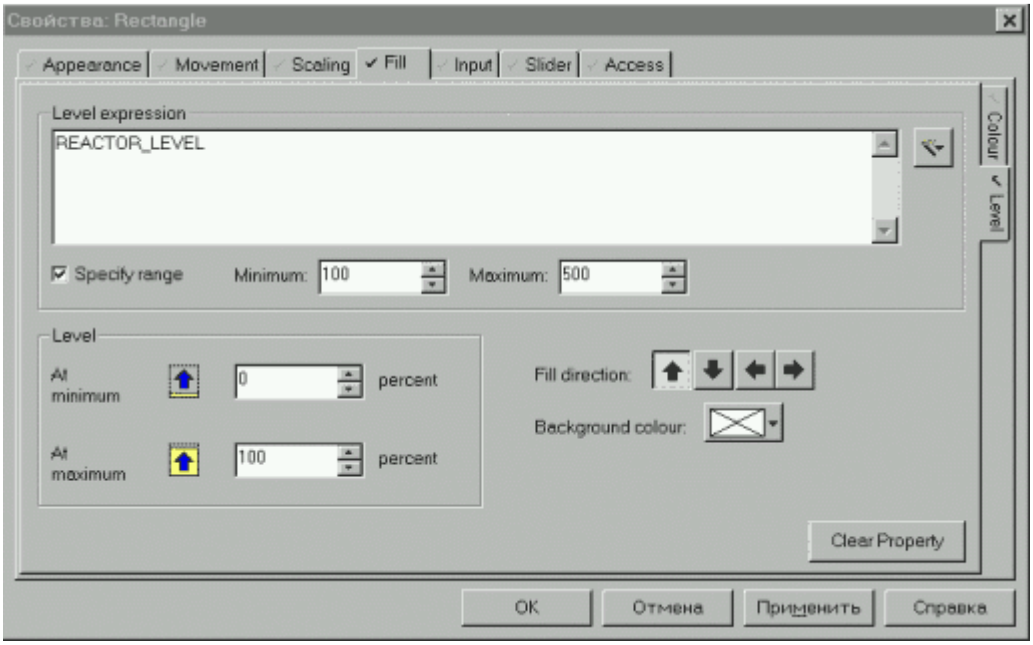


Рис.1.2.5. Диалог Свойства объекта с открытой закладкой Fill.

Закладка Input предоставляет разработчику возможность связать с объектом некоторую команду (поле Up Command, рис.1.2.6), которая будет выполняться при щелчке мышью на объекте. Можно также связать объект с командой, подаваемой с клавиатуры.

В диалоге Slider определяются объекты, которые можно использовать в качестве регуляторов. При перемещении объекта оператором (например, ползунка по шкале) значение соответствующей переменной будет меняться.

Название следующего диалога - Access (доступ) - говорит само за себя. Здесь определяется круг зон и объектов, доступных каждому из пользователей. Например, доступ к таким объектам, как регулятор, предоставляется не всем операторам, а только просмотр текущего состояния параметров процесса может быть доступен всем.

Каждая из рассмотренных закладок диалога Свойства объекта в свою очередь имеет боковые закладки. Например, диалог Movement (рис.1.2.4) имеет три боковые закладки, связанные с типом перемещения: горизонтальное (Horizontal), вертикальное (Vertical) и вращательное (Rotational). В диалоге Fill (рис.1.2.5) представлено две боковые закладки: Colour (цвет) и Level (уровень). Для других диалогов боковые закладки помогут задать такие свойства, как видимость, команды клавиатуры, команды, которые будут выполняться при нажатии на объект, и т. д.

При заполнении рассмотренных выше диалогов в них часто рекомендуется вводить имена переменных, используемых в проекте, и функции Cicode. С одной стороны, это занимает много времени, с другой - повышается вероятность допустить ошибку при вводе имени переменной или Cicode - функции. Во избежание этого в диалогах предусмотрена иконка , с помощью которой можно открыть список переменных проекта или список функций Cicode, соответствующих выбранному диалогу (рис.1.2.6).

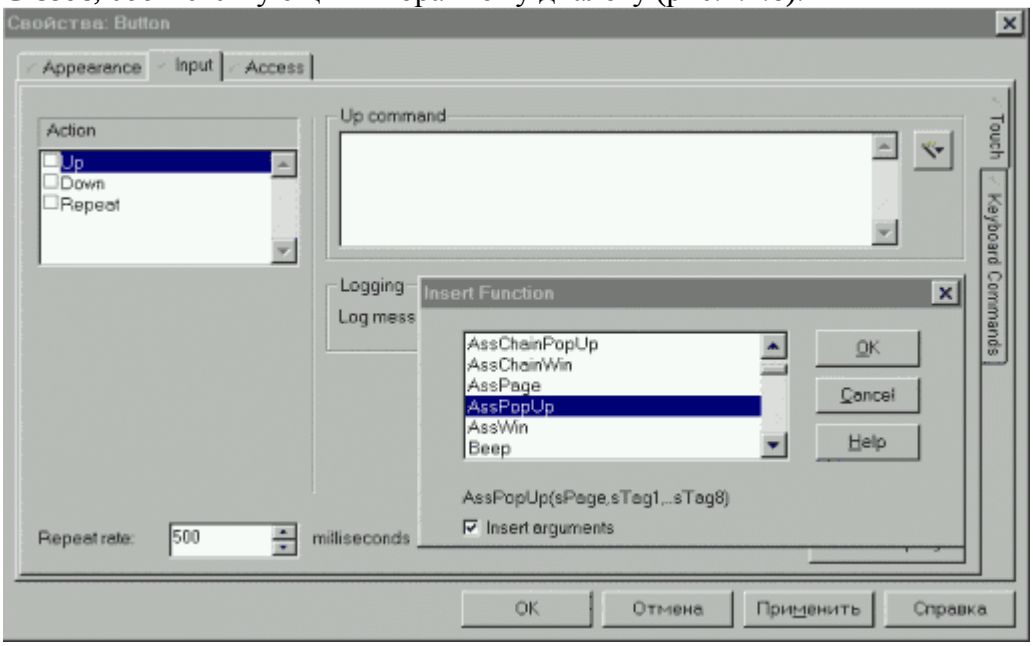


Рис.1.2.6. Диалог Свойства объекта на закладке Input со списком Cicode - функций.

Для вставки функции с аргументами следует отметить опцию Insert arguments. При этом выделенная инверсной подсветкой функция появится под окном списка функций с аргументами.

1.2.3. Bitmap Editor

Рисунок Bitmap представляет собой совокупность пикселей (точек). Этот рисунок может быть создан в специальном редакторе (Bitmap Editor) закрашиванием с помощью цветного карандаша ячеек сетки с разрешением в 1 пиксель. Рисунки Bitmap, как и обычные

объекты, можно перемещать, изменять их размеры, копировать, использовать как динамические объекты.

1.2.4. Библиотека статических объектов (Library Objects)

При разработке операторских интерфейсов пользователю приходится использовать графические объекты, представляющие собой технологические аппараты (колонны, емкости, теплообменники и т. д.), участки трубопровода, клапаны и такие агрегаты, как насос, электродвигатель, контроллер и т.д. Как правило, это сложные объекты, полученные объединением множества простых объектов или рисунки типа Bitmap.

Создание каждого из этих объектов требует большого времени и может значительно затянуть разработку приложения. Для ускорения работы над проектом Citect предлагает разработчику библиотеку объектов (Library Objects), которая включает более 500 готовых графических компонентов.

Библиотека состоит из большого количества разделов (например, раздел емкостей, теплообменников, клапанов, насосов, иконок и т. д.), каждый из которых содержит широкий набор объектов определенного типа.

На рис.1.2.8 представлен набор объектов раздела Емкости.

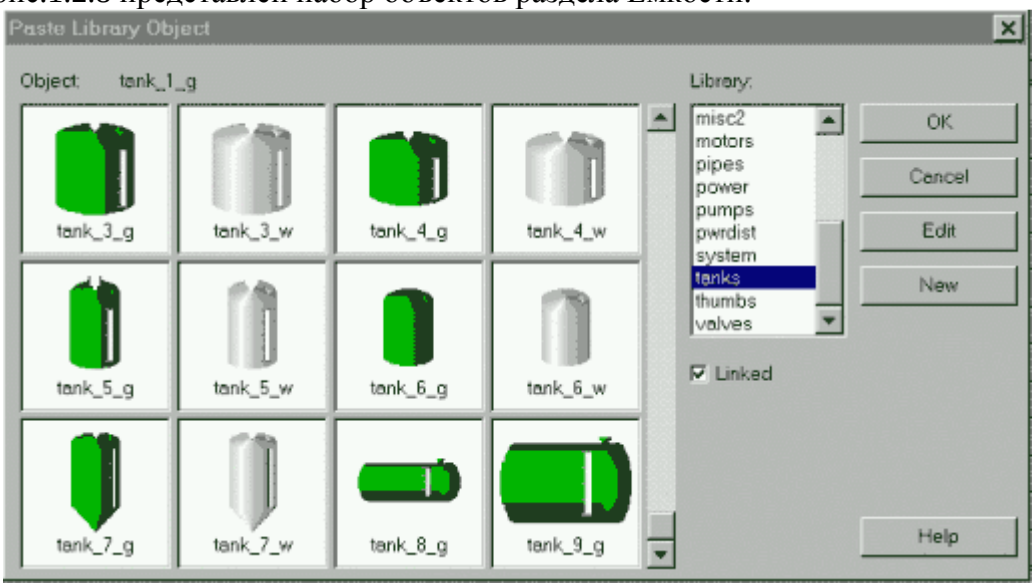


Рис.1.2.8. Раздел Емкости библиотеки объектов.

Если же нужного объекта в библиотеке нет, его можно импортировать из других Windows - программ. В Citect можно копировать объекты самых различных форматов: .BMP, .DXF, .PCX, .WMF и многих других. В крайнем случае, объект можно нарисовать и, чтобы в дальнейшем он всегда был "под рукой", скопировать в "свою" библиотеку.

1.2.5. Genies и Super Genies (джины и суперджины)

Каждый графический объект на графической странице настраивается индивидуально. Часто при разработке графического интерфейса приходится создавать типовые группы объектов, предназначенные для решения конкретной задачи. Например, группа из трех объектов (кнопка "ПУСК", кнопка "СТОП" и индикатор состояния - лампочка зеленого/красного цвета) предназначена для пуска/останова насоса, электродвигателя, конвейера и т. д. с индикацией их состояния. Тогда каждый раз для решения этой задачи разработчику придется создавать эти три объекта и конфигурировать их (задавать свойства). Но таких задач на одной графической странице может оказаться много. Читатель уже понял, что время специалиста в этом случае будет расходоваться неэффективно.

Для решения подобных задач Citect предлагает механизм, названный джином. Предлагается объединить несколько связанных задачей объектов в группу, предварительно придав этим объектам соответствующие свойства, а затем сохранить эту группу в библиотеке джинов, которая устроена аналогично библиотеке объектов. Джин может управляться как единый объект (его можно копировать, перемещать, масштабировать и т.д.), при этом обрабатываются все составляющие джина.

Теперь на решение вышеописанной задачи уйдет гораздо меньше времени. Надо лишь выбрать требуемого джина из библиотеки, вставить его в графическую страницу и в появившийся на экране диалог ввести имя/имена переменной/переменных.

Citect предлагает два типа сложных объектов:

- джины, которые размещаются на графической странице при проектировании системы, причем их количество на странице не ограничено;
- суперджины, которые представляют собой динамические страницы, активизируемые в режиме исполнения для ввода/вывода данных.

Таким образом, основное отличие этих двух механизмов в том, что джин объединяет несколько объектов и привязан к странице, а суперджин является отдельной страницей. В каких случаях может быть полезен суперджин ?

Когда технологические параметры поддерживаются на заданном значении контроллером (регулятором), оператор должен иметь возможность "вмешаться" в процесс (перейти с автоматического режима работы на ручной, изменить задание контроллеру). Постоянное нахождение на экране всех этих " рычагов " управления перегружает окно, к тому же пользоваться ими оператору приходится не часто. Вот тут и приходит на помощь суперджин (выпадающая страница).

Поскольку требования по управлению различными контурами регулирования идентичны, суперджин можно один раз создать, определив свойства его компонентов, и сохранить в библиотеку. Для использования суперджина на графической странице нужно лишь указывать его имя в команде, вызывающей этот суперджин на экран.

Объекты типа джин и суперджин позволяют экономить дисковое пространство компьютера, так как в его памяти хранится лишь одна копия.

Пакет Citect поставляется с библиотекой джинов и суперджинов. Вызов библиотеки производится автоматически при выборе инструмента Paste Genie (вставка джина). На рис.1.2.9 приведен раздел Моторы библиотеки джинов.

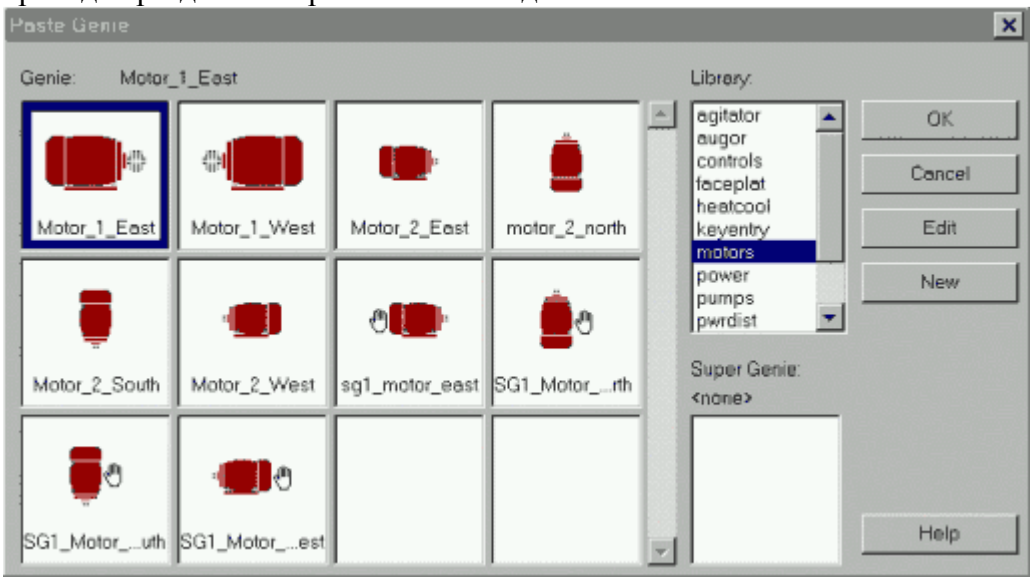


Рис.1.2.9. Раздел Моторы библиотеки джинов.

Часто суперджины и джины используются вместе. Это достигается привязкой джина к суперджину, когда одна из функций джина активизирует суперджин (выпадающую

страницу). В библиотеке джинов Citect некоторые джины уже связаны с суперджинами (джины с символом руки).

Механизм работы джина, связанного с суперджином, будет более понятен, если читатель внимательно прочтет следующие несколько абзацев. Следует еще раз подчеркнуть, что речь пойдет только о механизме "срабатывания" , а не о методике создания джинов и суперджинов.

Например, на мнемосхеме технологического процесса имеется несколько центробежных насосов. По каждому насосу оператор должен получать информацию о скорости вращения и иметь возможность управлять работой насоса: включить/выключить насос, выбрать режим ручного или автоматического управления насосом.

Задача очень простая - можно создать джин, реализующий все эти функции. Примерный вид этого джина представлен на рис.1.2.10а. На мнемосхеме представлено несколько насосов и для каждого нужен свой джин. Citect допускает любое количество джинов на странице, но она будет перегружена информацией, которая не нужна оператору постоянно.

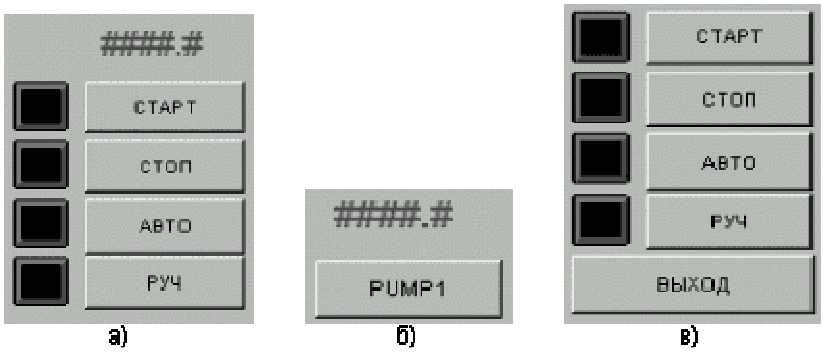


Рис.1.2.10. К описанию механизма суперджина.

Предлагается второе решение этой задачи - создать джин и суперджин. Постоянно на мнемосхеме процесса присутствуют джины для управления насосами, один из которых представлен на рис.1.2.10б. Но в этом случае они намного компактнее и не перегружают интерфейс. При определении свойств кнопки PUMP1 (насос1) джина на закладке INPUT (см. рис.1.2.6) в этом случае надо задать команду, которая будет выполняться при ее нажатии. Примером такой команды может быть AssPopUp (sPage, sTag1..8):

- AssPopUp - функция, открывающая суперджинн в выпадающей странице и имеющая следующие аргументы:
  - sPage - имя страницы суперджина;
  - sTag1..8 - имена переменных, связанных с суперджином.

Джин, связанный с суперджином (рис.1.2.10в), заранее создан и сохранен в библиотеку джинов и суперджинов. При определении свойств компонентов (кнопок) этого суперджина должны быть использованы заменяемые имена с синтаксисом ? type number ? :

- type - тип переменной (т.е. string, integer, real или digital);
- number - позиция имени переменной (1-8) в списке функции AssPopUp() джина, который вызывает страницу суперджина.

Применительно к рассматриваемому примеру функция (команда), вызывающая суперджин, может иметь вид AssPopUp("SGenie", "%PUMP%", "%STATUS%"), а заменяемые имена переменных суперджина - ?digital 1? и ?digital 2? .

Кнопки суперджина SGenie (рис.1.2.10в) имеют следующие свойства:

Закладка - Appearance (General) Опция - Text	**Закладка - Input (Touch) Поле - Command**
СТАРТ	? Digital 1?=1
СТОП	? Digital 1?=0
АВТО	? Digital 2?=1
РУЧ	? Digital 2?=0
ВЫХОД	WinFree( );

В режиме исполнения нажатие кнопки джина PUMP1 активизирует команду AssPopUp("SGenie", "%PUMP%", "%STATUS%"). При этом произойдет подстановка первой переменной PUMP в заменяемое имя ?digital 1?, а второй переменной STATUS - в заменяемое имя ?digital 2? суперджина, что вызовет появление выпадающей страницы с пультом управления насосом PUMP1 (рис.1.2.10в). После выполнения действий по управлению насосом выпадающая страница может быть закрыта щелчком по кнопке ВЫХОД (функция WinFree( ) закрывает страницу).

В результате применения суперджинов выигрывает оператор, который взаимодействует с управляемым процессом через интерфейс, имея всю необходимую информацию и средства управления.

1.3. Сравнение графических средств

Безусловно, графический интерфейс рассматриваемых систем достаточно многообразен. Но следует отметить, что подход к инструментарию различен.

- CiT - компания имеет большой опыт в разработке проектов. Взгляд CiT- это взгляд разработчика, который из опыта "ощущает", какие готовые решения следует предложить для ускорения и упрощения разработки мнемосхем технологического процесса.
- Wonderware - компания с большим опытом разработки инструментальных прикладных систем. Многим знакомый Windows - подобный интерфейс (по предлагаемым панелям, по способу создания окон) позволяет интуитивно использовать навыки работы в Windows. Библиотеки сложных графических объектов позволяют пользователю решать как вопросы дизайна (не все же художники !), так и сокращения времени разработки.
- Библиотеки Wizards в InTouch включают тысячи мастер-объектов. Воспользоваться Wizard - объектом просто хотя бы потому, что любое неправильное действие по его конфигурированию проверяется при закрытии каждого диалога. Если какое - то поле заполнено неправильно, то об этом предлагается информация с целью модификации неправильных параметров.
- Citect предлагает библиотеки символов, джинов и суперджинов. Использование всего арсенала названных средств предполагает:
    - концептуальное понимание;
    - заполнение диалоговых панелей свойств с целью анимирования сопровождается диагностикой лишь серьезных ошибок, а весь список ошибок появляется лишь на этапе компиляции проекта.
- Наконец, InTouch ориентирован на более широкий круг разработчиков операторских интерфейсов, так как он не предъявляет высоких требований к пользователю с точки зрения программирования. С этой работой после небольшой подготовки справится специалист-технолог или инженер по автоматизации технологических процессов.
- Citect предлагает более гибкий инструментарий, оставляя возможность пользоваться простыми средствами. Но соблазн велик! Для более полного использования возможностей Citect желательна более квалифицированная подготовка разработчиков приложений

ГЛАВА 2. ОРГАНИЗАЦИЯ ВЗАИМОДЕЙСТВИЯ С КОНТРОЛЛЕРАМИ

Современные SCADA - системы не ограничивают выбора аппаратуры нижнего уровня (контроллеров), 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.

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.

Изначально протокол DDE применялся в первых человеко - машинных интерфейсах 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 появятся 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.

2.1. 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.

Прикладные протоколы, используемые для организации взаимодействия с контроллерами, оставлены за границей этой книги.

2.2. Особенности построения коммуникационного программного обеспечения

Перед рассмотрением реализации связи с устройствами ввода/вывода в SCADA - системах InTouch и Citect читателю предлагается общий взгляд на организацию коммуникационного ПО в системах управления (рис.2.2.1).

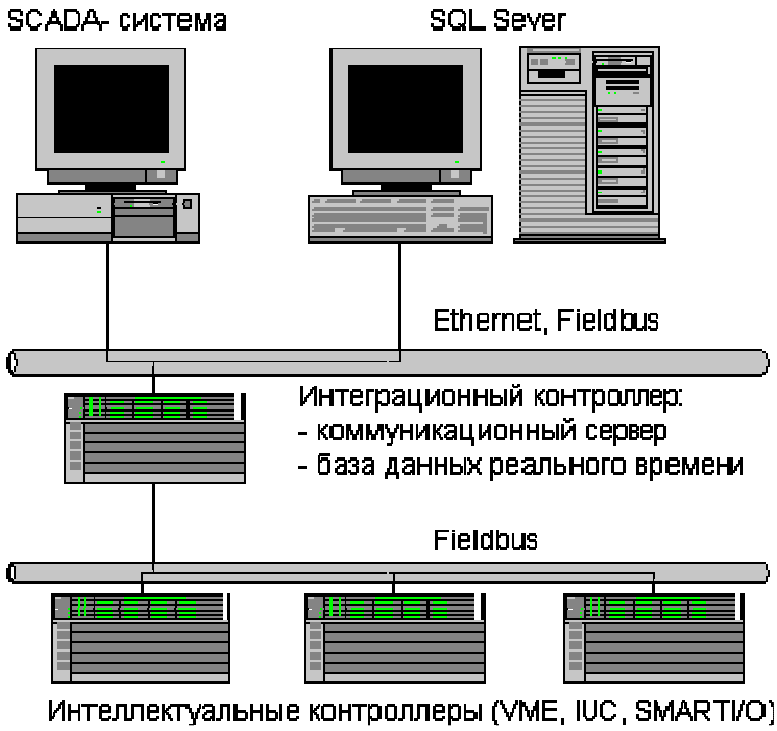


Рис. 2.2.1. Типовая архитектура системы управления.

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.2.2.2 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.

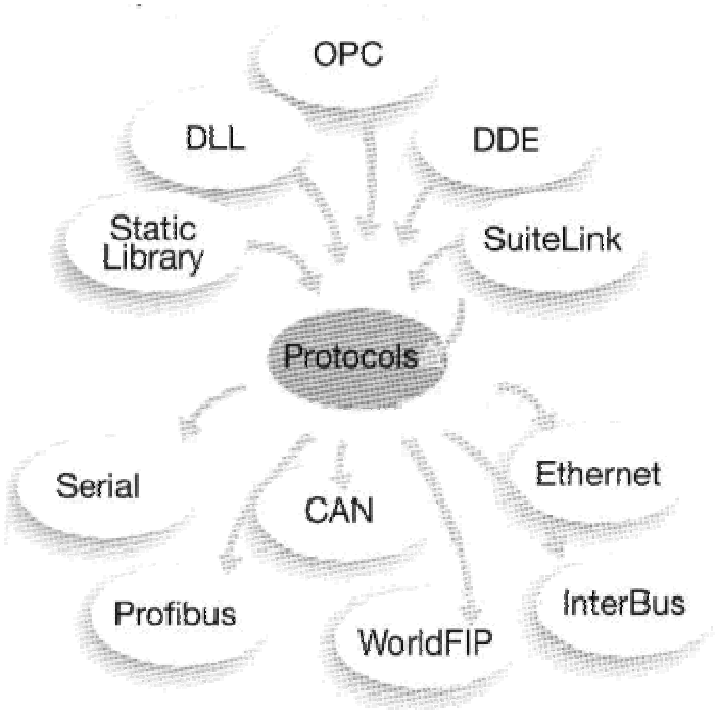


Рис. 2.2.2. Набор интерфейсов для SCADA - систем и спектр поддерживаемых протоколов.

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 вставная 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.

На рис.2.2.3 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.

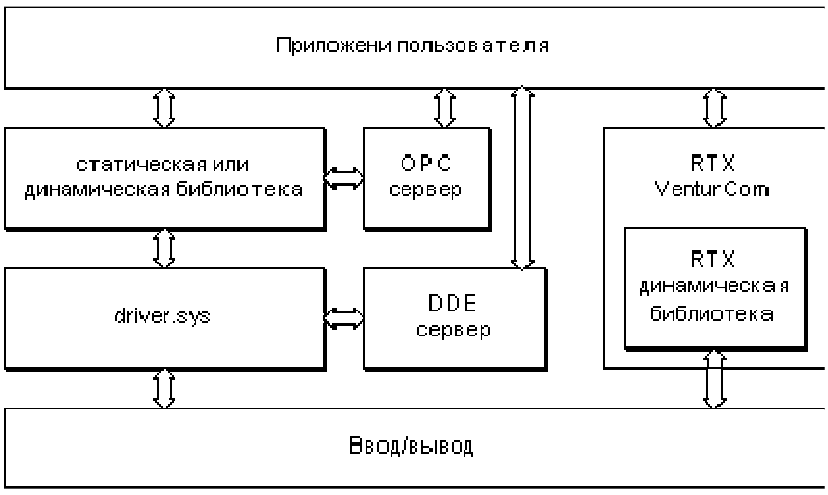


Рис.2.2.3. Схема организации коммуникационного ПО для Windows NT.

После рассмотрения общей схемы организации коммуникационного ПО представляется логичным остановиться на особенностях подключения к нему рассматриваемых в данной книге SCADA-приложений.

2.3. Серверы ввода/вывода в InTouch

При функционировании InTouch - приложения в реальном времени информация обо всех его переменных хранится в базе данных. К такой информации относятся имя переменной, ее тип, минимальное и максимальное значения, уставки, способ отображения (дисплей, журнал) и т. д., а также информация о коммуникационных каналах, по которым происходит обмен данными между технологическим процессом и приложением. InTouch - приложение поддерживает взаимодействие с DDE и OPC-серверами. Именно на организации взаимодействия с ними и остановимся ниже.

2.3.1. Поддерживаемые коммуникационные протоколы

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.

В InTouch поддерживается также пакетированный DDE - обмен - FastDDE. Применение последнего заметно повышает эффективность и производительность обмена данными благодаря уменьшению общего количества DDE - пакетов, которыми клиент и сервер обмениваются между собой. Но принципиальные недостатки, связанные с надежностью и зависимостью от количества загруженных в текущий момент приложений Windows, остались. Необходимость в появлении более совершенного технологичного протокола созрела! Но следует отметить, что отказ от DDE-механизма происходит не мгновенно хотя бы потому, что в мире наработано большое количество DDE - серверов.

С целью расширения возможностей стандартного протокола DDE на локальную сеть компания Wonderware предложила NetDDE. Он позволяет приложениям, запущенным на объединенных в локальную сеть компьютерах, вести DDE - обмен. Позднее NetDDE лицензируется компанией Microsoft и поставляется в дистрибутивном пакете Windows. Следует отметить и то, что NetDDE допускает обмен информацией между приложениями на IBM PC и приложениями на машинах другого типа с операционной системой VMS или UNIX. Компания Wonderware предлагает и инструментальные средства для разработки DDE-серверов, в том числе и для не-Windows-платформ.

Протокол SuiteLink был специально разработан фирмой Wonderware для того, чтобы удовлетворить таким требованиям, как целостность данных, высокая производительность и простота диагностики. В основе протокола SuiteLink лежит протокол TCP/IP. SuiteLink не является заменой протоколам DDE, FastDDE и NetDDE. Новый протокол разработан для поддержания быстродействующих промышленных систем и обладает следующими характеристиками:

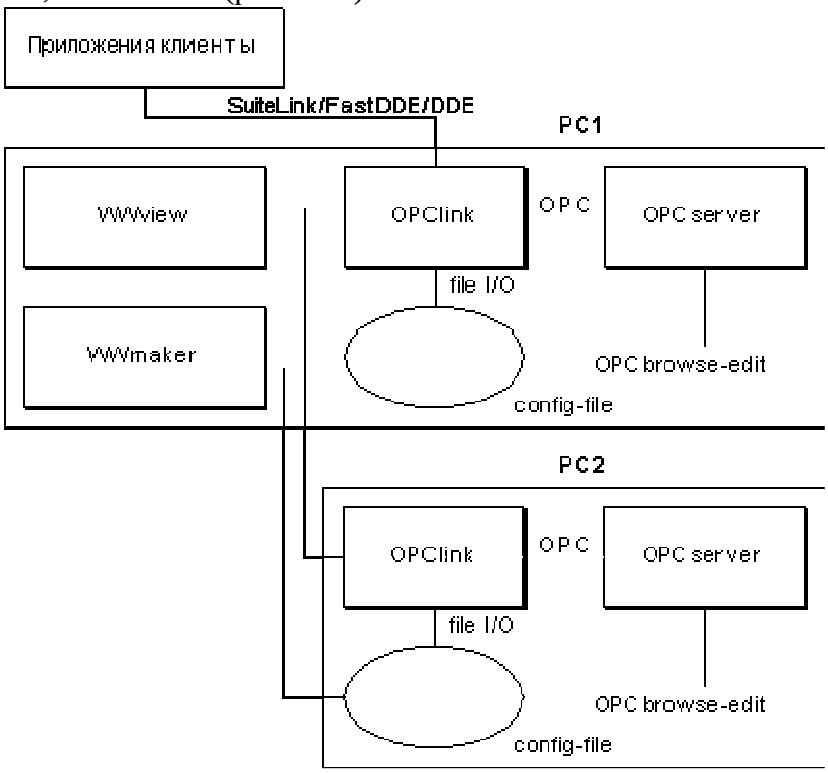
- Передача данных осуществляется в формате VTQ (Value, Time, Quality - значение, время, качество), в соответствии с которым каждая пересылаемая клиенту единица информации сопровождается метками времени и качества данных.
- Благодаря системному монитору операционной системы Windows NT (Performance Monitor) стал возможным расширенный анализ производительности по передаче данных, степени загрузки сервера, степени потребления ресурсов компьютера и сети, что особенно важно для проектирования и сопровождения больших распределенных промышленных сетей.
- Поддержка обмена данными между приложениями происходит независимо от того, исполняются ли эти приложения на одном узле сети или на разных.

Для реализации функций OPC - клиента Wonderware предлагает OPCLink - сервер, преобразующий OPC в SuitLink - протокол.

В материалах, предложенных компанией Wonderware, отмечается, что большинство реализованных OPC-серверов создают для каждого подключаемого к серверу клиента новый канал связи или нить. Для текущей обработки каждого клиента сервер должен переключаться между нитями. Каждая нить использует DCOM (Distributed Component Object Model) для организации обмена данными, и DCOM также управляет переключением нитей. В итоге возможна достаточно низкая производительность в сети.

Тесты, проведенные фирмой Wonderware, показали, что при обслуживании OPC-сервером 7 клиентов (при передаче 4 целых чисел в режиме обновления) сервер на 95% занимал ресурсы CPU. Это означает, что ресурсы компьютера практически целиком были заняты переключением нитей и DCOM- процедурами.

Поэтому на текущем этапе параметры производительности протокола SuiteLink превосходят параметры DCOM. Поставляемый в комплекте FactorySuite (Wonderware) OPCLink Server обеспечивает прием информации с OPC- сервера и передачу ее по протоколу SuiteLink в SCADA - систему InTouch и наоборот. Именно OPCLink Server рекомендуется устанавливать на одном узле с OPC- сервером, чтобы для сетевых передач использовался SuiteLink- протокол, а не DCOM (рис.2.3.1).



2.3.1. Использование SuiteLink - протокола в SCADA - системах.

Все описанные ниже особенности адресации распространяются и на OPC-серверы с одним лишь ограничением. При разработке InTouch - приложения создается канал связи с OPCLink - сервером (как с любым другим SuiteLink - сервером). Но рекомендуется использовать встроенный в InTouch OPC Browser для упрощения выбора параметров конфигурации подключаемого OPC - сервера.

2.3.2. Особенности адресации в InTouch

В InTouch вышеуказанные механизмы положены в основу обмена данными между приложениями InTouch и DDE и SuiteLink - серверами, которые, в свою очередь, связаны коммуникационными каналами с устройствами нижнего уровня (контроллерами).

Так как InTouch предназначен для разработки и поддержания интерфейса сбора данных и диспетчерского управления (рис.2.3.2), среда исполнения WindowViewer при

взаимодействии с контроллерным уровнем выступает, как правило, в роли приложения - клиента (узел View), запрашивающего данные у приложения - сервера (I/O Server).

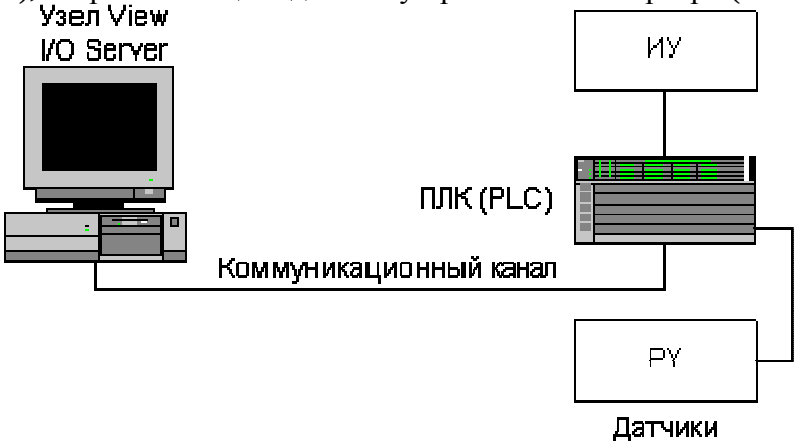


Рис.2.3.2. Обмен данными между InTouch - приложением и технологическим процессом.

Через сервер ввода/вывода InTouch - приложение имеет возможность читать данные из контроллера или писать данные в него. Процесс обмена информацией InTouch - приложения с контроллером можно представить следующей схемой (рис. 2.3.3).

Рис. 2.3.3. Схема обмена информацией InTouch - приложения с ПЛК.

Здесь и встает один из главных вопросов организации обмена с серверами ввода/вывода: каким образом обеспечить клиенту доступ к запрашиваемой им информации?

Для организации обмена с приложением определяются каналы обмена или каналы доступа, характеризующиеся следующими параметрами:

- имя узла (Node Name);
- имя приложения (Application Name);
- имя группы данных или топик (Topic Name);
- имя элемента (Item Name).

Имя приложения - это имя программы Windows, которая выполняет функции DDE, FastDDE, SuiteLink - серверов. Имя группы данных (топика) определяется при конфигурировании сервера на прием или передачу группы данных, которыми сервер будет обмениваться с контроллером или объединенными в сеть контроллерами. Определенные параметры группы (топика) зависят от конкретного сервера (поэтому рекомендуется изучать документацию и справочную систему выбранного сервера). Например, при использовании Modbus - сервера, позволяющего обеспечить взаимодействие с контроллером Modicon Micro 984 PLC, в качестве имени приложения (Application Name) должен быть Modbus, в качестве имени группы или топика (Topic Name) вводится любое имя (текстовая строка), но среди необходимых параметров группы из списка выбирается имя контроллера Modicon 984 PLC. А в качестве имени элемента (Item Name) следует выбирать название конкретного регистра контроллера (например, 40001 для контроллера Modicon Micro 984). Чтобы узнать правильный синтаксис имени элемента, необходимый для конкретных PLC, нужно обратиться к руководству по соответствующему серверу.

Определены все компоненты коммуникационного канала. С учетом введенных понятий схема обмена информацией для рассмотренного выше примера будет выглядеть следующим образом (рис.2.3.4).

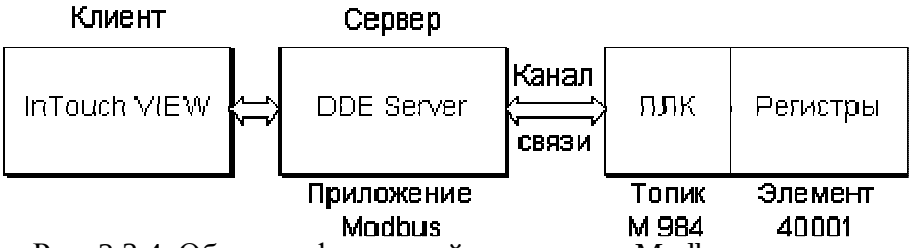


Рис. 2.3.4. Обмен информацией на примере Modbus - сервера.

Фирма Wonderware предлагает DDE и SuiteLink - серверы, которые поддерживают более 800 типов контроллеров основных производителей и различные протоколы.

Если нужного драйвера все-таки нет, можно воспользоваться пакетом разработки драйверов FactorySuite Toolkit.

Схемы, приведенные на рис. 2.3.3, 2.3.4, интерпретируют стандартный обмен информацией между узлом (приложением) View и контроллером (ПЛК) в режиме сбора данных и управления. В этом режиме, как уже было сказано выше, приложение View - клиент по определению.

2.3.3. Обмен данными с другими приложениями

Но приложения InTouch могут взаимодействовать не только между собой, но и с другими Windows - приложениями. Одним из известных примеров такого приложения является Microsoft Excel. InTouch - приложение может считывать и записывать какие - либо значения в любую клетку открытой в Excel электронной таблицы. Аналогично и программа Excel может читать и записывать информацию в базу данных InTouch - приложения. Данный механизм обеспечивает одновременное обновление данных в одном приложении при изменении их значений в другом.

Если клиентом (приложением, запрашивающим информацию) по - прежнему является узел View, то Excel - это приложение, поставляющее информацию (сервер). В качестве группы или топика (Topic) тогда будет выступать имя таблицы Excel, а элемент обмена информацией - ячейка в таблице Excel (табл.2.1, вариант 1). Когда клиентом является приложение Excel, а сервером - приложение View, группой в этом случае всегда является словарь переменных InTouch (база данных) с именем Tagname. Элементом обмена будет элемент базы данных - имя переменной (табл.2.1, вариант 2).

Таблица 2.1.

Приложение-клиент	Приложение-сервер	Группа	Элемент
View	Excel	Sheet1.XLS	R1C1
Excel	View	Tagname	R_Level

В случае обмена данными по сети с использованием пакета Wonderware NetDDE необходимо к трехуровневой структуре адреса добавить четвертый уровень - имя удаленного узла сети (Node Name).

Подводя итог вышесказанному, следует подчеркнуть, что информация по доступу к данным устройств ввода/вывода или других приложений должна храниться в приложении (в словаре переменных). И разработчику в InTouch-приложении важно подключиться к вышеописанному каналу доступа. Для этого в InTouch необходимо определить имя доступа Access Name и связать его с переменной приложения.

2.3.4. Определение имени доступа в словаре переменных InTouch

В InTouch - приложениях вся информация о переменных приложения хранится в Tagname Dictionary (Словарь переменных). Это не что иное, как база данных реального времени - один из центральных компонентов InTouch.

При определении переменной в базе данных InTouch запрашивает определенную информацию о каждой переменной, например, имя переменной, ее тип, имя доступа и т. д. В пакете InTouch используется два базовых типа переменных - Memory (внутренние) и I/O (переменные ввода/вывода). Переменные типа Memory могут быть использованы для создания различных системных констант, моделирования элементов системы управления и в вычисляемых переменных, доступных другим Windows - программам. Все переменные, которые получают или передают свое значение другой Windows - программе, должны иметь тип ввода/вывода (I/O). В эту категорию попадают переменные, которые посредством канала доступа (Access Name) принимают или отправляют данные из/в серверов ввода/вывода, других приложений InTouch, других программ Windows.

Определение новой переменной в базе данных InTouch, как и просмотр, и модификация атрибутов уже существующих переменных, производится в диалоге Tagname Dictionary (рис.2.3.5). Доступ к этому диалогу осуществляется командой Special/Tagname Dictionary в окне среды разработки WindowMaker или двойным щелчком по иконке Tagname Dictionary в окне Application Explorer.

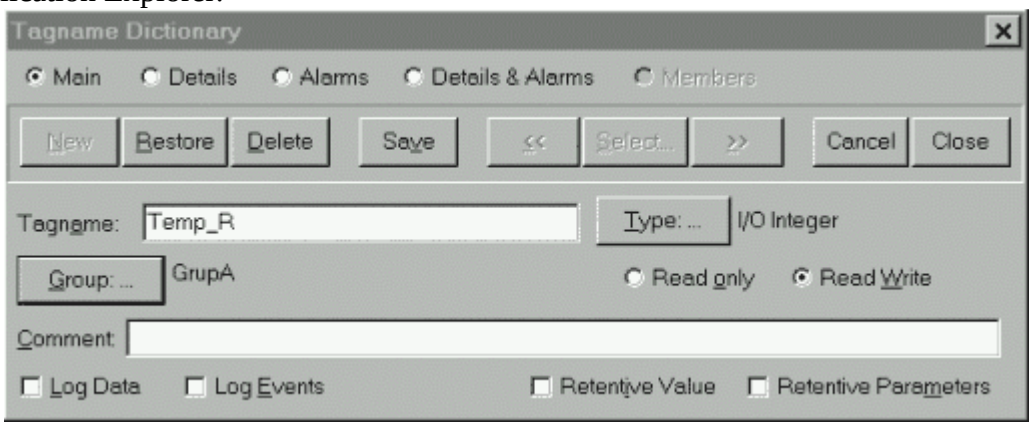


Рис. 2.3.5. Диалог Tagname Dictionary (Словарь переменных).

Поля Tagname и Comment предназначены для ввода имени переменной и соответствующего комментария. По умолчанию включена опция Read/Write (чтение/запись). Можно отметить и опцию Read Only, если в процессе исполнения WindowViewer должен только читать значение переменной. В любое время в режиме проектирования можно открыть список переменных приложения щелчком по кнопке Select для выбора соответствующей переменной, просмотра списка или модификации атрибутов. Диалог Select Tag (выбор переменной) представлен на рис.2.3.6.

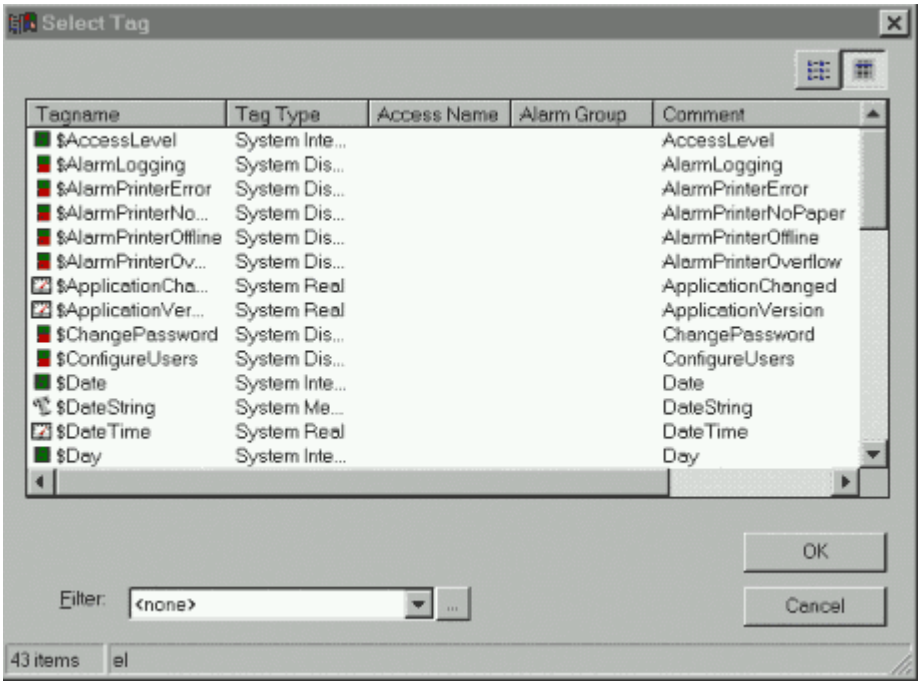


Рис. 2.3.6. Диалог Select Tag (выбор переменной).

Для каждой переменной в этом диалоге приведена следующая информация: имя переменной, ее тип, имя доступа, группа аларма и комментарий. Группа алармов (Alarm group, рис.2.3.6) для переменной определяется в диалоге, вызываемом нажатием кнопки Group диалога Tagname Dictionary. Все, что касается алармов, рассматривается в соответствующем разделе ниже. Выбор типа переменной осуществляется в диалоге Tag Types (тип переменной, рис. 2.3.7), вызываемом на экран нажатием кнопки Type диалога Tagname Dictionary.

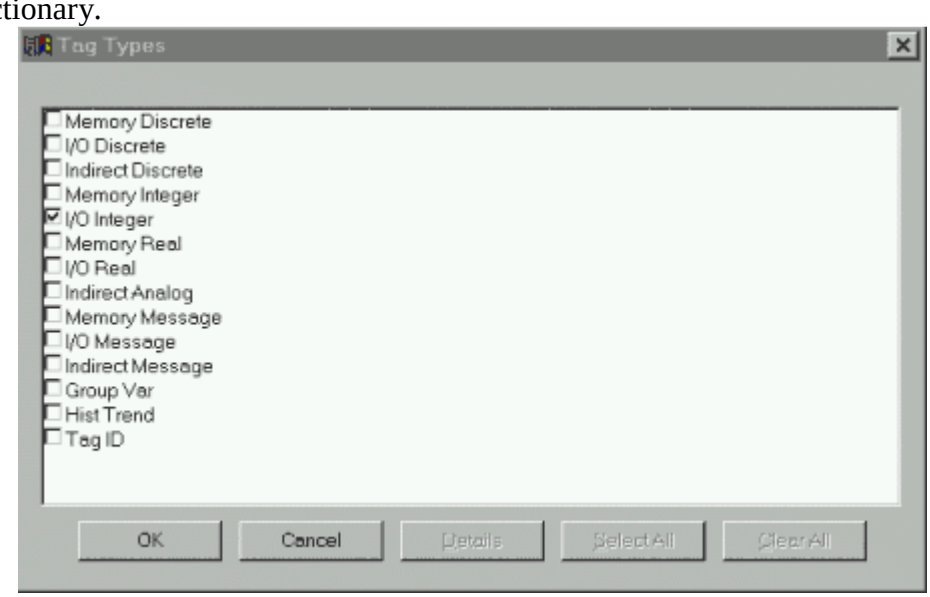


Рис. 2.3.7. Диалог Tag Types (тип переменной).

В этом диалоге представлен полный список основных типов переменных InTouch. Выбор завершается отметкой соответствующей опции и щелчком по Ok. После выбора типа переменной программа возвращает пользователя в диалог Tagname Dictionary (Словарь переменных). При этом будет открыт и дополнительный диалог подробного описания переменной, содержание которого зависит от выбранного типа. На рис.2.3.8 представлен диалог подробного описания вещественной переменной типа I/O Integer.

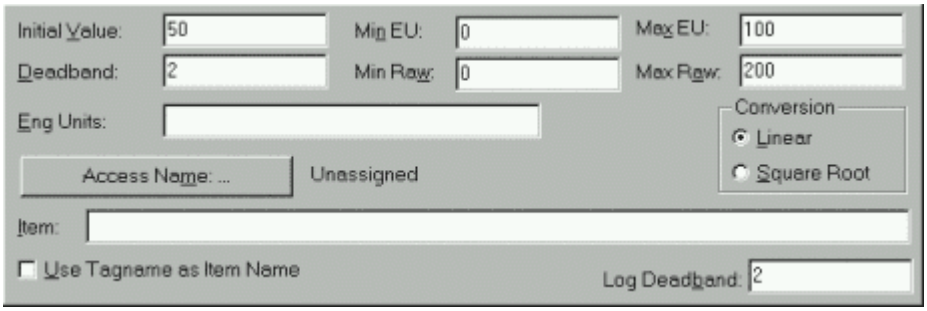


Рис. 2.3.8. Диалог подробного описания переменной типа I/O.

Кнопка Access Name (имя доступа) используется для определения канала обмена (канала доступа) с сервером, с которым будет связана описываемая переменная. Имя доступа Access Name определяется именем узла, именем приложения и именем группы или топика. Имя топика должно совпадать с соответствующим именем, заданным при конфигурировании DDE, SuiteLink-сервера. Имя элемента, как компонента многоуровневого адреса, определяется в поле Item (рис.2.3.8).

В распределенных системах InTouch имя доступа может быть определено либо как локальный адрес, либо как глобальный.

Локальные адреса используются в том случае, когда View - узлы имеют свои серверы ввода/вывода. На рис. 2.3.9 узлы исполнения (View - узлы), каждый со своей копией одного и того же приложения, ссылаются на свои собственные источники данных ввода/вывода (серверы ввода/вывода).

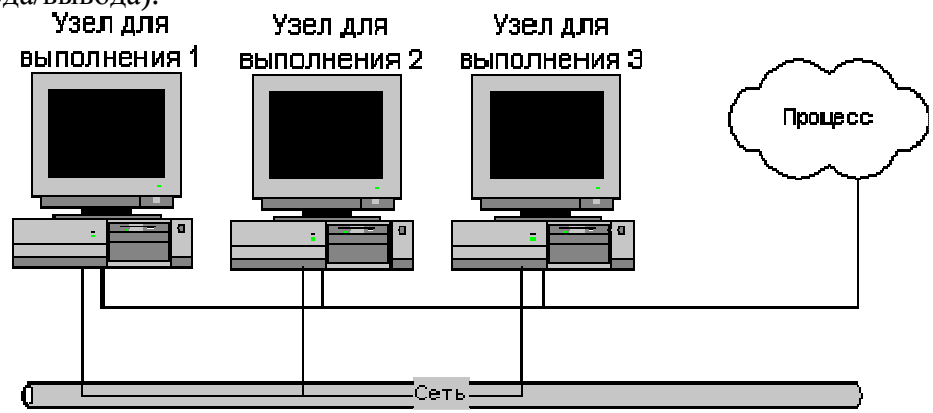


Рис. 2.3.9. Сеть View - узлов с собственными серверами ввода/вывода.

Поэтому при определении канала доступа к информации ввода/вывода достаточно трехуровневого адреса (Application - приложение, Topic - объект, Item - элемент). Имя узла (Node) в этом случае опускается. Щелчок по кнопке Access Name (рис.2.3.8) вызывает на экран одноименный диалог.

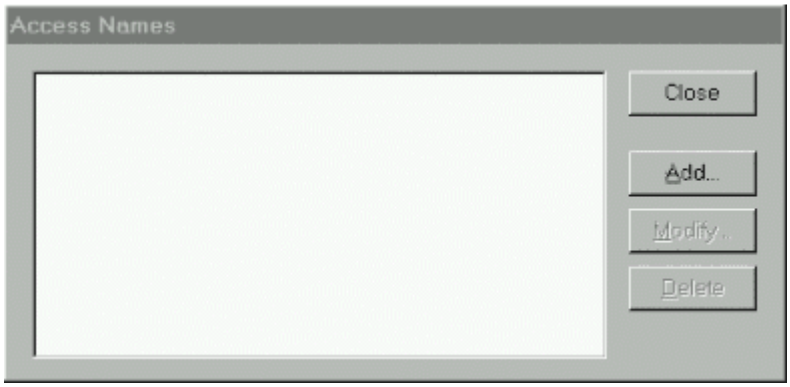


Рис. 2.3.10. Диалог Access Names (имена доступа).

Этот диалог предназначен для определения нового канала доступа (кнопка Add), модификации существующего (Modify) или удаления (Delete). Щелчок по кнопке Add вызывает диалог определения нового канала доступа (рис.2.3.11).

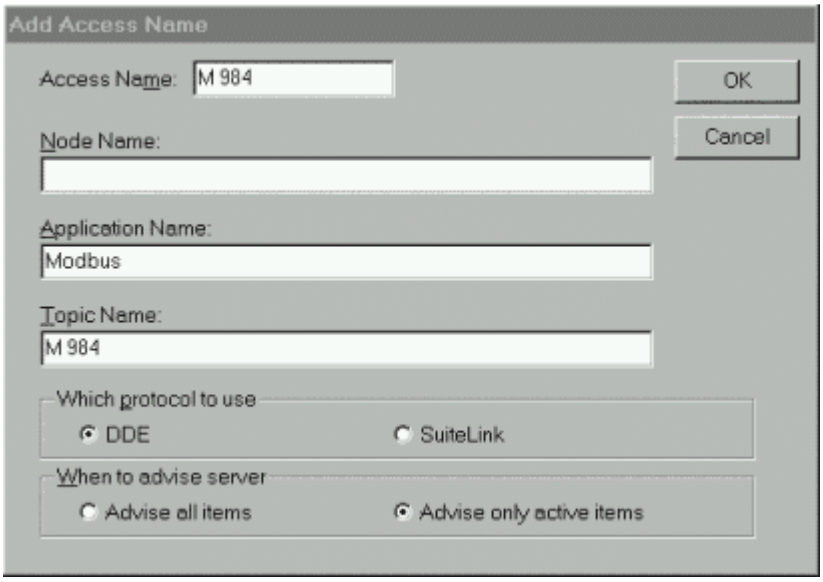


Рис.2.3.11. Диалог определения нового канала доступа (локальный адрес).

Диалог определения канала доступа заполнен в соответствии с примером, рассмотренным на рис.2.3.4. В качестве имени (канала) доступа (Access Names) рекомендуется выбирать имя группы или топика (Topic Name). Следует подчеркнуть, что поле Node Name (имя узла) оставлено пустым. Щелчок по кнопке Ok возвращает пользователя в диалог Access Names (имена доступа) с определенным именем доступа (рис.1.2.12).

Рис.2.3.12. Диалог Access Names с определенным именем доступа.

Глобальные адреса источников данных ввода/вывода позволяют нескольким View - узлам обращаться к одному и тому же серверу ввода/вывода. Такой подход предоставляет возможность отказаться от нескольких серверов ввода/вывода, однако менее защищен от отказов (рис.2.3.13).

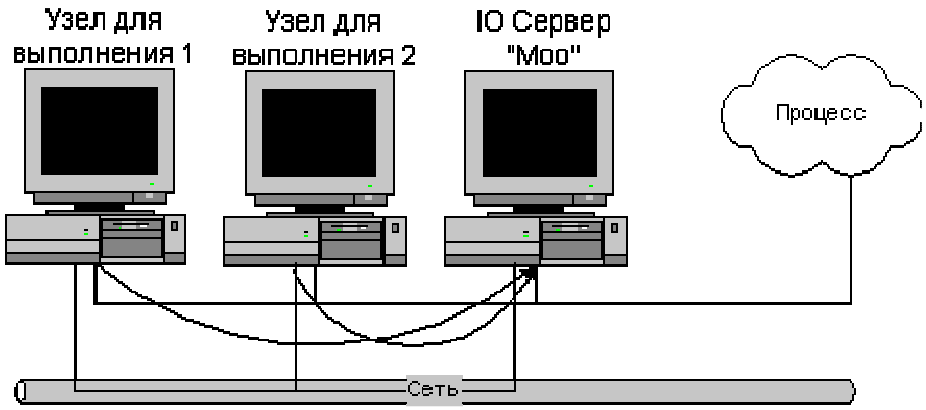


Рис.2.3.13. Архитектура с двумя View - узлами и сервером ввода/вывода.

Два View - узла исполняют идентичные копии одного и того же приложения и ссылаются на один и тот же источник ввода/вывода (I/O сервер). Поэтому при определении канала доступа к информации ввода/вывода необходимо использовать четырехуровневый адрес (Node - узел, Application -приложение, Topic - объект, Item - элемент). Заполненный диалог при определении имени доступа для такой конфигурации представлен на рис. 2.3.14.

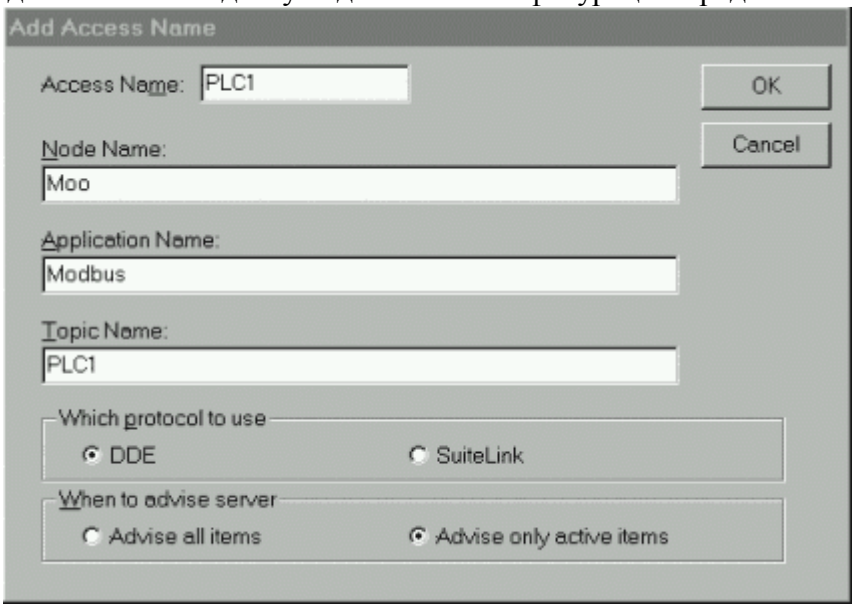


Рис.2.3.14. Диалог определения нового канала доступа (глобальный адрес).

При выборе имени доступа действует то же правило, что и при локальной адресации: рекомендуется, чтобы это имя совпадало с именем группы данных или топика (Topic Name). Но поле Node Name (имя узла) необходимо заполнить. В качестве этого имени при глобальной адресации выбирают имя узла, на котором установлен сервер ввода/вывода, являющийся источником данных для нескольких приложений.

Для каждой переменной ввода/вывода задается атрибут Access Name. С одним именем доступа, как правило, связано большое количество переменных. Распределение переменных по группам (топикам) - произвольное. Но для оптимизации функционирования серверов рекомендуется в одну группу относить переменные с одинаковой частотой обновления. В противном случае частота, задаваемая при конфигурировании топика в сервере, должна соответствовать минимальному временному кванту. Желательно на этапе конфигурирования сервера определить группы (топики) для каждого частотного диапазона и в соответствии с этими группами создать имена доступа (Access Name) в InTouch (лучше даже, чтобы имена групп совпадали с именами доступа). А далее каждую описываемую в InTouch-приложении переменную типа I/O связывать с подходящим именем доступа для обеспечения рационального пакетирования данных.

2.4. Коммуникационные возможности в Citect

2.4.1. Коммуникационные протоколы

Для обмена данными с контроллерами в Citect могут использоваться следующие способы: встраиваемые драйверы, DDE - обмен, OPC - протоколы.

- Первый путь предполагает создание динамических библиотек, выполняющих функцию драйверов. Citect поставляется с более чем 120 драйверами ввода/вывода. Все эти драйверы 32 - разрядные и обеспечивают подключение более 300 типов ПЛК, RTU, микроконтроллеров, Loop - контроллеров и т. д. Среди них контроллеры фирм ABB (AC 110, AC 160, AC 410, AC 450, Commander 100, 150, 200, 300), Advantech (Adam 4000, Adam 5000), Allen Bradley (PLC-5, PLC-5/250, PLC-2, PLC-3, SLC 500), Bristol Babcock (33xx RTUs), Control Microsystems (TeleSAFE), Fuji, Foxboro (760 Series), GE Fanuc (Series 90, Series 9070, Series 9030, Series 6), Hewlett Packard (HP 3852A), Hitachi (H20, H200, H250, H700), Honeywell (620 Series, TDC2000, UDC3000), Koyo (405 Series), Mitsubishi (Melsec A, AnA, FX), Modicon (Series 484, Series 584, Series 884, Series 984), Motorola (Moscad RTU), Omron, Samsung (Fara PLC), Siemens (Simatic - модели S5, S7, TI), Toshiba (EX 100, EX 250, EX 500, EX 2000, Tosdic-200, DPCS, PCS, OIS, SIS), Yokogawa (4082 Hybrid Recorder, 3880 Hybrid Recorder, Micro XL, Centum XL) и многих других фирм. Если нужного драйвера в системе Citect не окажется, можно воспользоваться пакетом разработки драйверов Driver Development Kit (DDK).
- Связь через DDE - сервер использует стандартный коммуникационный протокол Windows. Citect поддерживает связь с любым DDE - сервером.
- Система Citect может функционировать в качестве и OPC - сервера и OPC - клиента.

2.4.2. Установка связей с устройствами ввода/вывода

Система Citect имеет в своем составе специальную утилиту - Express Communications Wizard (система установки связи) - средство быстрого и простого конфигурирования устройств. Эта программа использует полученную на каждом шаге процесса установки информацию и снабжает разработчика установками по умолчанию, оставляя в тоже время варианты выбора параметров ввода/вывода. Каждый диалог программы содержит четыре кнопки управления процессом установки связи:

- Next - продолжение установки;
- Back - возврат на предыдущий шаг;
- Cancel - отмена установки;
- Help - справочная информация.

Щелчок по кнопке Finish последнего диалога завершает установку связи. Доступ к системе установки связи осуществляется в Citect Explorer из папки Communications соответствующего проекта (рис. 2.4.2).

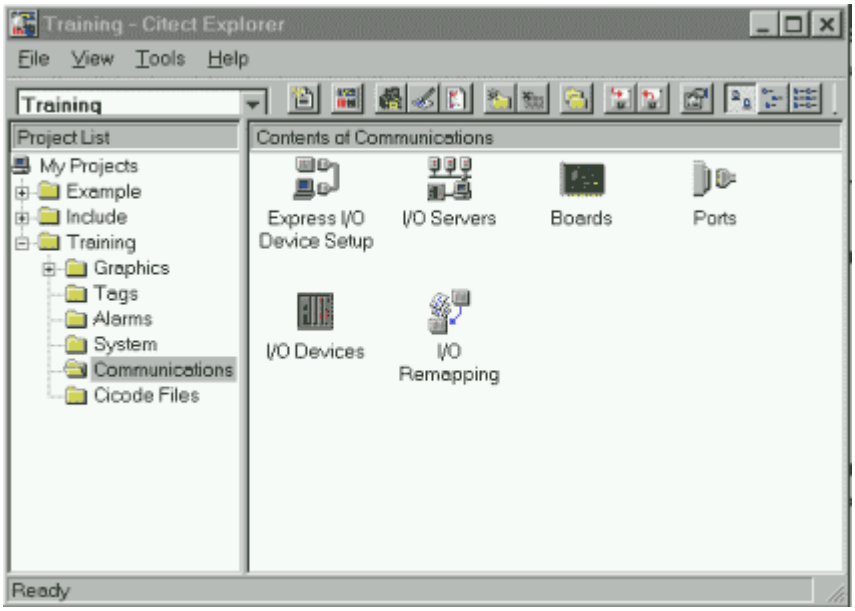
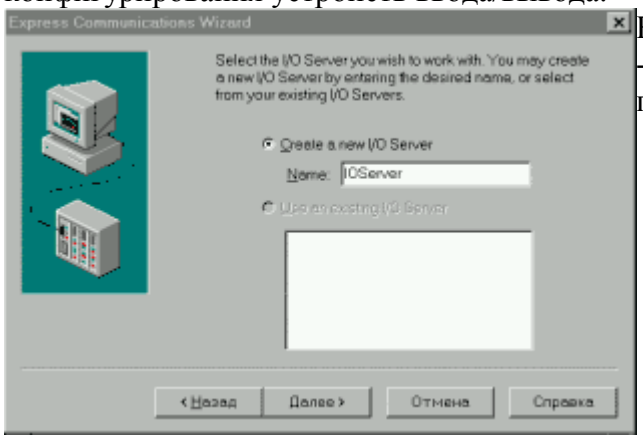


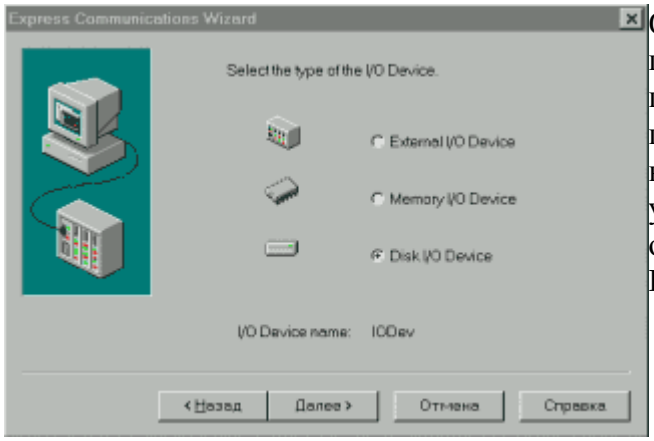
Рис. 2.4.2. Доступ к мастеру коммуникаций из Citect Explorer.

Двойной щелчок по иконке Express I/O Device Setup запускает процесс установки и конфигурирования устройств ввода/вывода.



В этом диалоге предлагается определить Citect -компьютер как сервер ввода/вывода и присвоить ему уникальное имя.

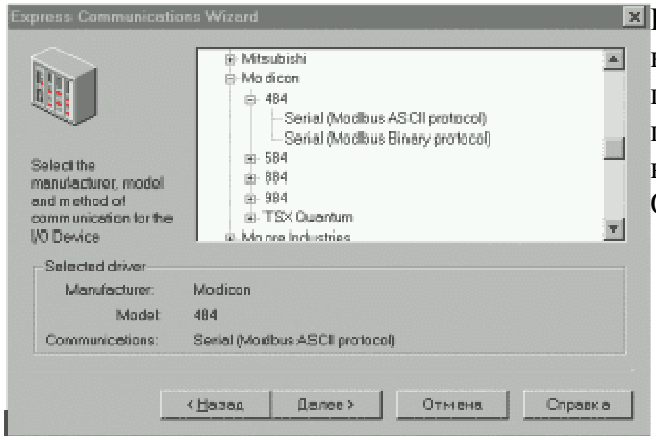
Последовательное нажатие клавиши Next (далее) открывает перед разработчиком новые диалоги, предлагая ввести необходимую информацию по установке связи между Citect и устройством ввода/вывода.



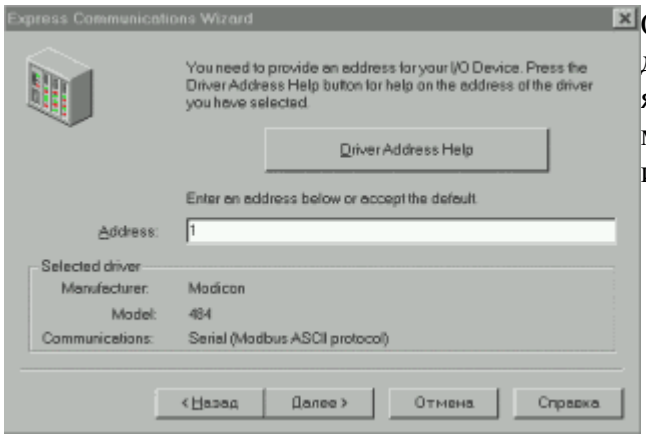
Citect предоставляет возможность пользователю разрабатывать и отлаживать проект без необходимости физического подключения к реальному устройству ввода/вывода. Просто при конфигурировании устройства ввода/вывода его можно определить как внутреннее (Memory I/O Device) или как диск (Disk I/O Device).

Теперь Citect будет работать так, как будто взаимодействует с реальным контроллером. При выборе Disk I/O Device данные сохраняются в виде файла на жестком диске. При перезапуске Citect данные остаются доступными. Disk I/O Device может использоваться и

другими компьютерами через ЛВС (LAN). Данные, записанные в Memory I/O Device, теряются при перезапуске системы.



В этом диалоге производится выбор марки контроллера, интерфейсной платы и протокола обмена информацией. Для обмена по OPC-протоколу именно в этом диалоге выбирается протокол OPC, чтобы наделить Citect-приложение функциями OPC-клиента.



Одним из основных элементов при обмене данными между компьютером и устройством является адрес устройства. Эту информацию можно найти в документации на используемый сервер ввода-вывода.

В результате работы Express Communications Wizard будет заполнено несколько диалогов, полностью характеризующих установленную связь между Citect- компьютером и устройством ввода/вывода. Находясь в Citect Explorer (см. рис. 2.4.2), можно дважды щелкнуть по соответствующей каждому диалогу иконке и отредактировать параметры связи.

Диалоги, автоматически заполненные в процессе работы Express Communications Wizard при установке связи между Citect - компьютером и контроллером Mitsubishi Melsec-FX Series PLC, подсоединенным к последовательному порту Com1, показаны на рис. 2.4.3.

- В диалоге Server (сервер) для определения сервера задают его имя в поле Server Name. При наличии двух серверов (дублирование) каждый сервер должен иметь свое имя.
- Диалог Boards (интерфейсная плата) включает следующие поля:
  - имя сервера (Server Name);
  - имя интерфейсной платы (Boards Name);
  - тип интерфейсной платы (Boards Type);
  - адрес интерфейсной платы (Address);
  - адрес порта в интерфейсной плате (I/O port).

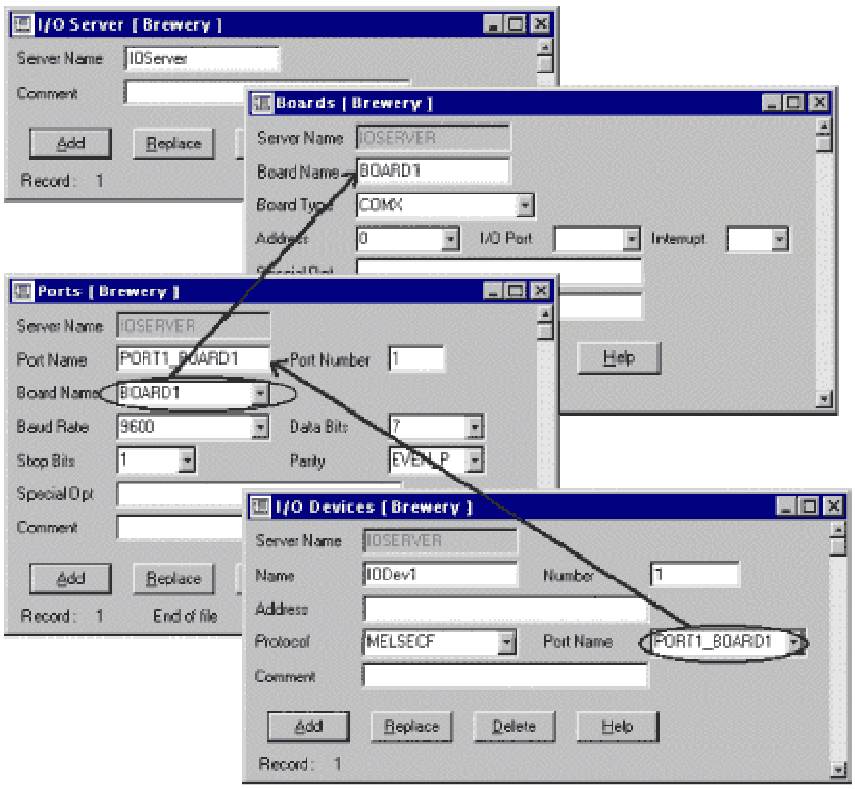


Рис. 2.4.3. Диалоги конфигурирования параметров связи.

- Диалог Ports (порт) включает следующие поля:
  - имя порта (Port Name);
  - номер порта (Port Number);
  - имя интерфейсной платы (Boards Name);
  - скорость в бодах (Baud Rate);
  - количество битов (Data Bits) - 7 или 8;
  - количество стоповых битов (Stop Bits) - количество битов в конце посылки (1 или 2);
  - контроль на четность (Parity).
- Диалог I/O Device (устройство ввода/вывода) включает следующие поля:
- имя устройства ввода/вывода (Name);
- номер устройства ввода/вывода (Number) - 0 - 4095;
- адрес (Address); - протокол (Protocol) - большинство устройств поддерживает ряд протоколов, выбор которых зависит от выбранного метода связи;
- имя порта (Port Name), обеспечивающего взаимодействие с устройством ввода/вывода.

Итак, канал связи полностью определен, и это заняло у опытного пользователя всего несколько десятков секунд (в крайнем случае, пару минут). Теперь предлагается определить переменные, подключаемые к этому каналу связи. Находясь в Citect Explorer, следует открыть папку Tags, а затем дважды щелкнуть на иконке Variable Tags. На экране появится диалог (рис.2.4.4).

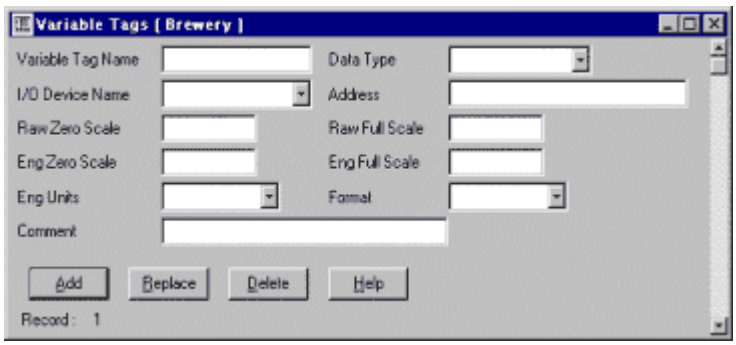


Рис. 2.4.4. Диалог Variable Tags (переменная).

Для каждого переменной следует определить:

- уникальное имя (Variable Tag Name);
- тип данных (Data Type);
- имя устройства ввода-вывода (I/O Device Name);
- адрес (Address);
- формат данных (Format) и т. д.

Этот диалог придется заполнять для каждой переменной, нажимая каждый раз клавишу Add (добавить). Хотя информация, вводимая по каждой переменной, достаточно однотипна, при большом количестве переменных процесс будет достаточно трудоемким.

Все переменные проекта хранятся в формате DBF, и возможно непосредственное редактирование баз данных с использованием таких программных продуктов, как Microsoft Excel. Файл с базой данных Variable.dbf находится в директории \Citect\User\. Такая возможность работы с базой данных переменных позволит существенно сократить сроки разработки проекта. Фрагмент файла Variable.dbf приведен на рис. 2.4.5.

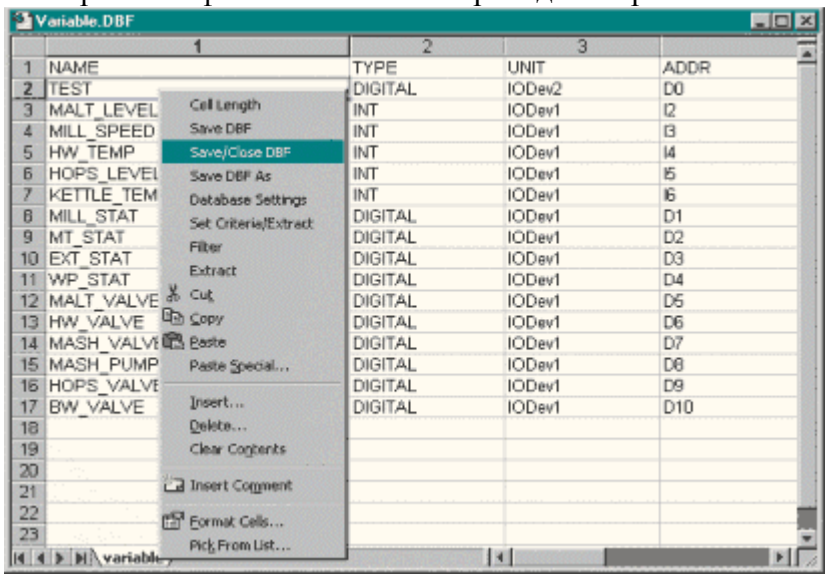


Рис. 2.4.5. Фрагмент базы данных в таблице Excel.

2.5. Подключение узлов Citect

2.5.1. Архитектура клиент – сервер

Citect ориентирован на реализацию архитектуры клиент - сервер и имеет в своем составе пять функциональных модулей (серверов или клиентов):

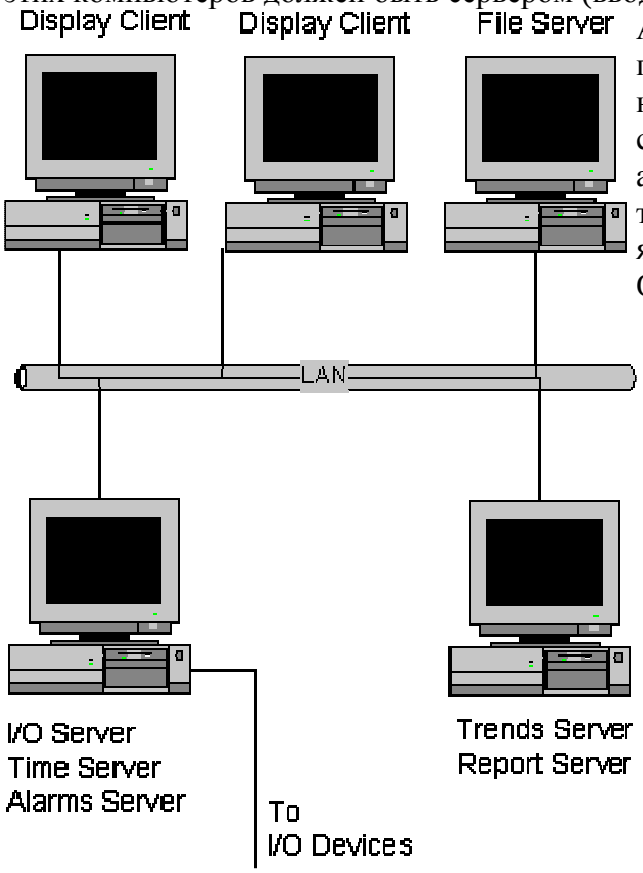
- I/O - сервер ввода/вывода. Обеспечивает передачу данных между физическими устройствами ввода/вывода и другими модулями Citect.

- Display - клиент визуализации. Обеспечивает операторский интерфейс: отображение данных, поступающих от других модулей Citect, и управление выполнением команд оператора.
- Alarms - сервер алармов. Отслеживает данные, сравнивает их с допустимыми пределами, проверяет выполнение заданных условий и отображает алармы на соответствующем узле визуализации.
- Trends - сервер трендов. Собирает и регистрирует трендовую информацию, позволяя отображать развитие процесса в реальном масштабе времени или в ретроспективе.
- Reports - сервер отчетов. Генерирует отчеты по истечении определенного времени, при возникновении определенного события или по запросу оператора.

Каждый функциональный модуль Citect исполняется как отдельная задача независимо от того, исполняются ли модули на одном компьютере или на разных. Поэтому Citect позволяет строить архитектуры различной сложности. Простейшая архитектура системы Citect состоит из одного компьютера, на котором работают все модули. Очевидно, эта архитектура применима лишь для малых систем с ограниченным числом параметров.

Для средних и больших проектов (тысячи и десятки тысяч параметров) можно использовать сетевые возможности Citect. Компьютеры системы управления могут быть распределены по всему предприятию (цехам, участкам, офисам) и поставлять информацию оперативному персоналу и различным службам.

Сетевые возможности Citect допускают использование в локальной сети до 256 компьютеров. Каждый из них может играть роль Display Client. Но по меньшей мере один из этих компьютеров должен быть сервером (ввода/вывода, алармов, трендов, отчетов).



Архитектура клиент - сервер может быть представлена разнообразными вариантами: например, один компьютер может быть сервером ввода/вывода данных и сервером алармов, другой - сервером отчетов и сервером трендов. Остальные компьютеры сети являются клиентами визуализации - Display Client (рис.2.5.1- слева).

File Server - компьютер с большой емкостью памяти (жесткий диск, лазерные диски) для хранения всей информации локальной сети (сервер базы данных). Для очень больших систем можно предложить вариант, в котором каждая задача обслуживается отдельным компьютером (сервер ввода/вывода, сервер тревог, сервер трендов и сервер отчетов), причем

клиентами визуализации могут быть несколько компьютеров. Пример такой системы приведен на рис. 2.5.2.

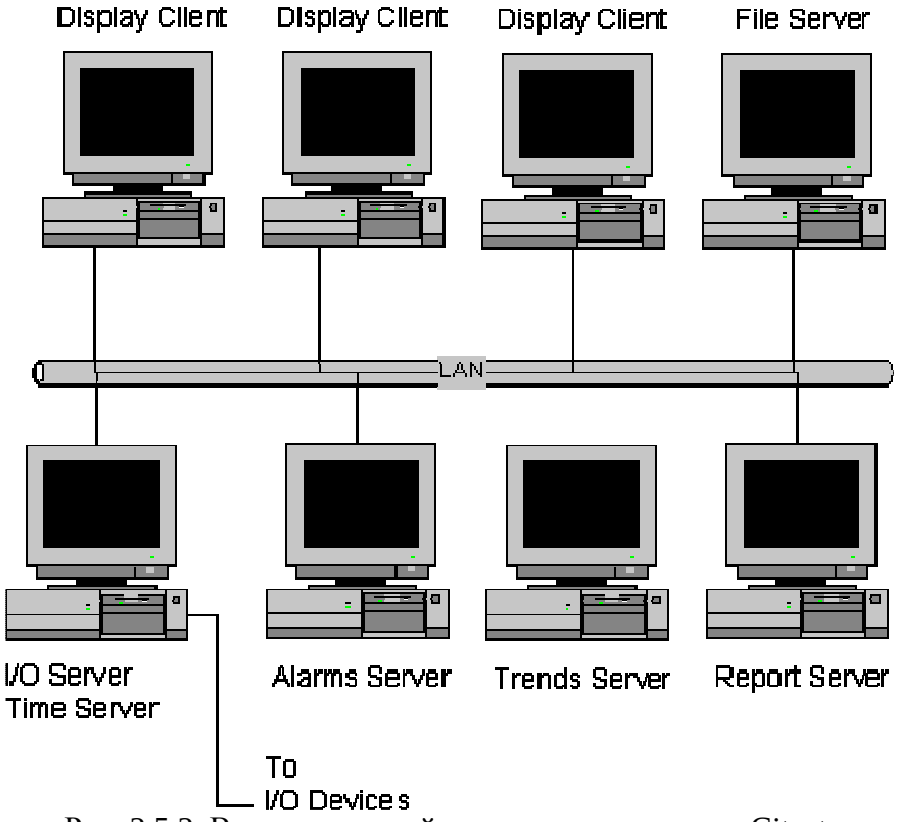


Рис. 2.5.2. Вариант сетевой архитектуры системы Citect.

Следует отметить, что для рассматриваемых архитектур можно использовать только один сервер алармов, сервер трендов и сервер отчетов. В то же время допускается использование нескольких серверов ввода/вывода (I/O Server).

2.5.2. Конфигурирование Citect-компьютеров в сети

Сетевые средства Citect построены на базе NetBIOS и поддерживаются такими сетевыми протоколами, как NetBEUI, IPX/SPX, TCP/IP. С другой стороны, Citect поддерживает все сетевые протоколы, совместимые с NetBIOS, что существенно расширяет спектр сетей, с которыми может взаимодействовать Citect. К таким сетям можно отнести Ethernet, Arcnet, Internet, Novell Netware, LAN Manager и др.

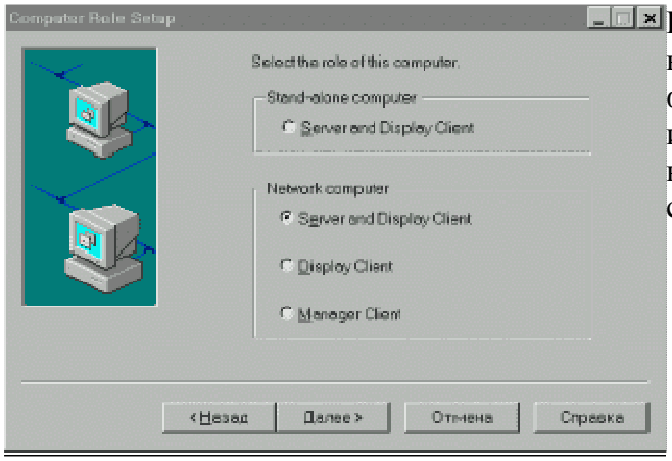
Конфигурирование Citect-компьютеров в локальной сети, т. е. распределение клиент-серверных задач между узлами системы управления, производится с помощью системы конфигурирования компьютеров (Computer Setup Wizard), входящей в состав системы Citect.

При этом пользователю не предлагается выбор протокола для сетевого обмена. Citect по умолчанию использует протокол NetBEUI.

Запуск Computer Setup Wizard производится в Citect Explorer. Для этого следует сначала щелкнуть по строке списка проектов, а затем еще раз щелкнуть по иконке Computer Setup.

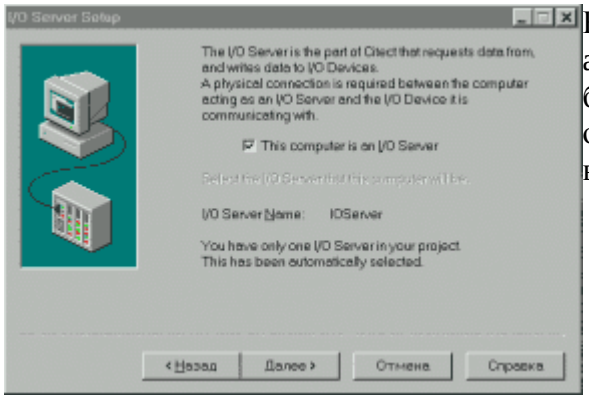
Первый диалог предлагает выбрать режим работы программы. Работа Computer Setup Wizard может производиться как в экспресс - режиме (рекомендуемые параметры), так и в режиме выборочной установки (пользовательские параметры).

Последовательное нажатие клавиши Next (далее) открывает перед разработчиком новые диалоги, предлагая ввести необходимую информацию по конфигурированию Citect-компьютера в сетевой архитектуре.



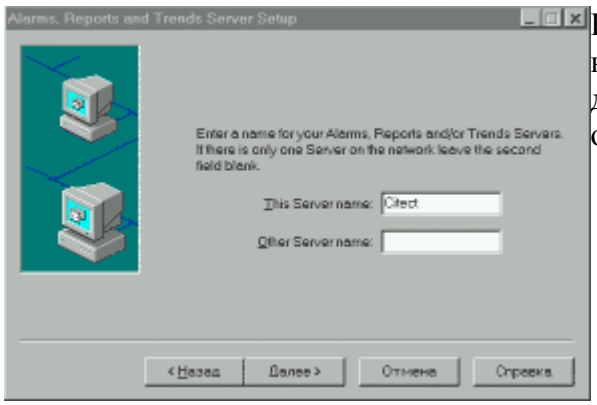
В этом окне определяется роль данного узла в Citect - системе: компьютер, выполняющий функции клиента и сервера, только клиента или мониторинговые функции. Здесь же надо определить, является ли компьютер сетевым или автономным.

При конфигурировании узла в сетевой архитектуре как Display Client (клиент визуализации) или Manager Client (компьютер с мониторинговыми функциями) следующие диалоги предложат разработчику определить имя сервера, к которому будет обращаться за информацией этот компьютер, имя компьютера для его идентификации в сети, а также настройки компьютера.

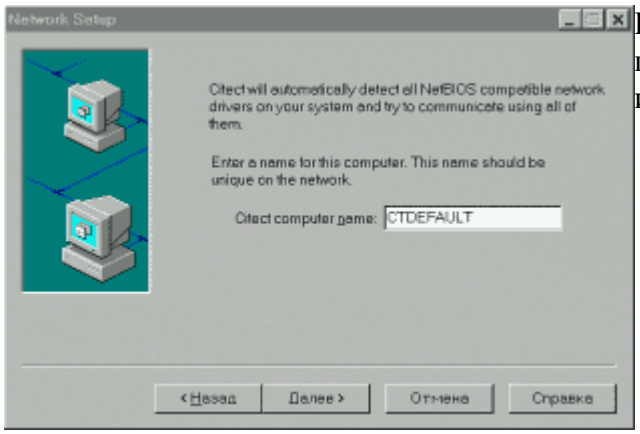


Если в предыдущем диалоге выбрать опцию Server and Display Client в сетевом применении, то далее будут открываться диалоги с предложениями определить этот компьютер, как сервер ввода/вывода, сервер алармов, трендов, отчетов.

Следует напомнить, что в локальной сети допускается использование нескольких серверов ввода/вывода, но только один компьютер может играть роль сервера алармов (сервера трендов или отчетов).



Если компьютер определен как сервер, ему необходимо присвоить имя. При наличии в сети другого сервера, его имя также должно быть отражено в диалоге.



Каждому из компьютеров сети следует присвоить уникальное имя для его идентификации в сети.

Таким образом, в результате проведенной работы по конфигурированию компьютеров сети для каждого из них должны быть определены следующие параметры:

- идентификационное имя в сети;
- сетевые функции (сервер, компьютер оператора, компьютер менеджера);
- имя каждого сервера;
- доступ к событиям;
- начальные настройки.

Почти все компоненты системы управления, созданной на базе Citect, могут быть дублированы: система отображения, серверы алармов, трендов, отчетов, сервер ввода/вывода, внешние устройства ввода/вывода (PLC), сетевые кабели, сетевой сервер базы данных и т. д. В зависимости от требований по надежности, предъявляемых к компонентам системы управления, при конфигурировании Citect-компьютеров их следует определять как основные или резервные.

2.6. Сравнение коммуникационных возможностей

Что же реально сегодня предлагают потребителю Wonderware и Ci Technologies в области коммуникаций? С точки зрения протоколов обе системы поддерживают DDE и OPC-обмены.

- Для улучшения характеристик DDE-обмена компания Wonderware предлагает пакетированный DDE, называемый FastDDE и свой протокол SuiteLink, обеспечивающий максимальную производительность по сравнению с DDE, FastDDE, OPC. Компания Ci Techologies поставляет встроенные драйверы, тем самым сводятся к нулю протокольные издержки.
- Citect-приложение может выполнять функцию не только OPC-клиента, но и OPC-сервера, что расширяет возможности Citect при построении различных конфигураций проектов.

С точки зрения организации взаимодействия между приложениями на различных узлах в сети следует указать на различие подходов компаний Wonderware и Ci Technologies.

- При разработке InTouch-приложения не важно, происходит ли подключение к серверу ввода-вывода или к переменным InTouch-приложения на другом узле. В обоих случаях единообразным способом описываются каналы доступа, определяются имена доступа и к ним привязываются переменные приложения. И в качестве протоколов обмена используются выбираемые при определении Access Name DDE или SuiteLink - протоколы.
- В Citect с помощью системы установки связи Express Communications Wizard можно определить только каналы обмена с устройствами ввода-вывода. Для организации обмена между Citect-приложениями (на разных узлах в сети) предлагается конфигурировать каждый узел с Citect-приложением на выполнение заданных функций (сервера ввода/вывода по отношению к другим Citect-узлам, серверов алармов трендов и т. д.).

ГЛАВА 3. АЛАРМЫ И СОБЫТИЯ

Состояние тревоги, в дальнейшем аларм (Alarm) - это некоторое сообщение, предупреждающее оператора о возникновении определенной ситуации, которая может привести к серьезным последствиям, и потому требующее его внимания, а часто и вмешательства. А принял - ли оператор сообщение об аларме? Чтобы снять эти сомнения, в системах управления принято различать неподтвержденные и подтвержденные алармы. Аларм называется подтвержденным после того, как оператор отреагировал на сообщение об аларме. До этого аларм оставался в состоянии неподтвержденного. Наряду с алармами в SCADA - системах существует понятие событий. События представляют собой обычные статусные сообщения системы и не требуют реакции оператора. Обычно событие генерируется при возникновении в системе определенных условий (типа регистрации оператора в системе). От эффективности подсистемы алармов зависит скорость идентификации неисправности, возникшей в системе, или технологического параметра, вышедшего за установленные регламентом границы. Быстродействие и надежность этой подсистемы могут существенно сократить время простоя технологического оборудования. Например, если оператор не получит вовремя информацию о том, что двигатель насоса перегрелся, это может привести в лучшем случае к выходу насоса из строя, а то и к крупной аварии.

Причины, вызывающие состояние аларма, могут быть самыми разными. Неисправность может возникнуть в самой SCADA-системе, в контроллерах, каналах связи, в технологическом оборудовании. Может выйти из строя датчик или нарушатся его метрологические характеристики. Параметры технологического процесса могут выйти за границы, установленные регламентом и т. д.

3.1. Типовые алармы

Подсистема алармов - это обязательный компонент любой SCADA - системы. Но возможности подсистем алармов различных SCADA - систем, вероятно, разные. С другой стороны, когда речь идет о типах алармов, то все SCADA - системы поддерживают такие типы алармов, как дискретные и аналоговые.

Дискретные алармы срабатывают при изменении состояния дискретной переменной. При этом для срабатывания аларма можно использовать любое из двух состояний: TRUE / ON (1) или FALSE / OFF (0). По умолчанию дискретный аларм может срабатывать на ON или OFF, в зависимости от конкретной SCADA - системы.

Аналоговые алармы базируются на анализе выхода значений переменной за указанные верхние и нижние пределы. Аналоговые алармы могут быть заданы в нескольких комбинациях:

- High и High High (верхний и выше верхнего);
- Low и Low Low (нижний и ниже нижнего);
- Deviation (отклонение от нормы);
- Rate of Change - ROC (скорость изменения).

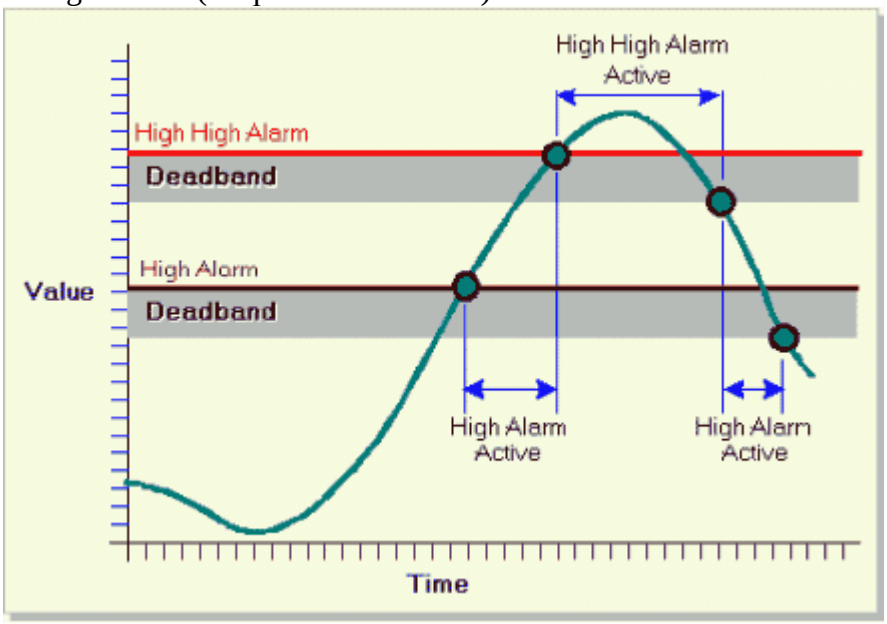


Рис.3.1.1. Графическая интерпретация алармов типа Hi и HiHi.

Из рис. 3.1.1 видно, что алармы Hi и HiHi срабатывают при достижении переменной заданных для каждого аларма пределов (High Alarm, High High Alarm). Для выхода переменной из состояния аларма (HiHi или Hi) необходимо, чтобы ее значение стало меньше порогового на величину, называемую зоной нечувствительности (Deadband). Аналогично можно интерпретировать алармы типа Lo и LoLo.

Все вышеизложенное справедливо и для аларма типа Deviation (рис.3.1.2), только речь в этом случае идет об отклонении значения переменной от заданного значения (Setpoint), причем это заданное значение в ходе технологического процесса может изменяться либо оператором, либо программно (автоматически). Аларм сработает при выходе значения переменной за границу предельно допустимого отклонения.

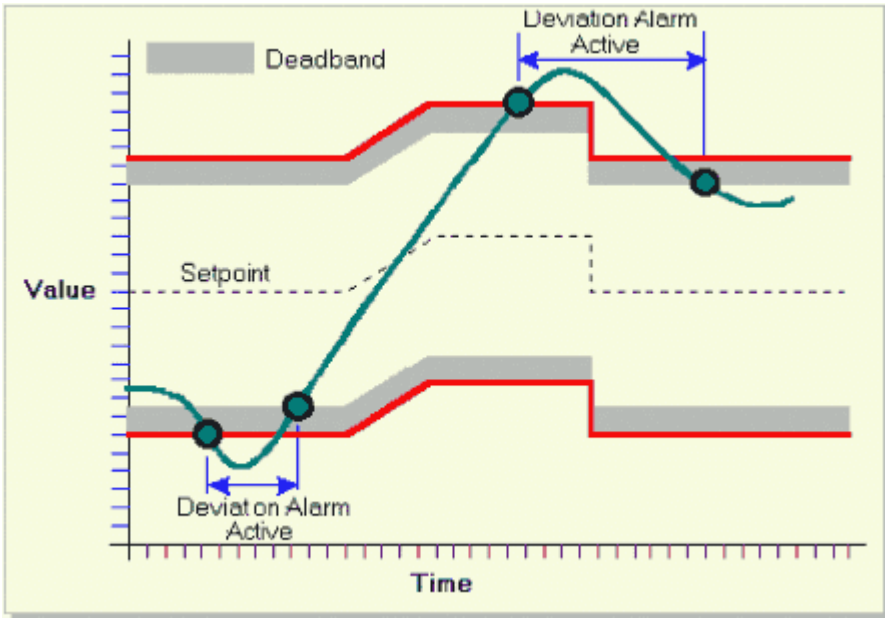


Рис.3.1.2. Графическая интерпретация алармов типа Deviation.

Алармы типа ROC срабатывают, когда скорость изменения параметра становится больше предельно допустимой. Понятие "зона нечувствительности" (Deadband) к алармам этого типа не применяется.

3.2. Алармы и события в InTouch

В InTouch имеется две системы алармов: стандартная и распределенная.

Стандартная система используется для отображения информации и подтверждения всех аварийных ситуаций и событий, возникающих в локальном InTouch - приложении.

Распределенная система расширяет возможности стандартной и позволяет подтверждать аварийные ситуации, генерируемые системами алармов других включенных в сеть InTouch - приложений.

InTouch поддерживает возможность отображения, регистрации и печати информации как об алармах, связанных с аналоговыми или логическими переменными, так и о системных событиях.

3.2.1. Типы алармов и событий

В зависимости от своих характеристик алармы подразделяются на несколько категорий по типу (Type) и классу (Class). Представление о типах и классах стандартной и распределенной систем можно получить из табл. 3.1.

Алармы	Стандартный тип	Распределенный класс	Распределенный тип
Discrete	DISC	DSC	DSC
Deviation - Major	LDEV	DEV	MAJDEV
Deviation - Minor	SDEV	DEV	MINDEV
Rate - of - Change	ROC	ROC	ROC
SPC	SPC	SPC	SPC
Value - LoLo	LOLO	VALUE	LOLO
Value - Lo	LO	VALUE	LO
Value - High	HI	VALUE	HI
Value - HiHi	HIHI	VALUE	HIHI

Таблица 3.1.

С InTouch - переменной можно связывать алармы любого типа. В зависимости от типа переменной для нее можно определять один или более классов и типов алармов.

События в InTouch также делятся в зависимости от их характеристик на несколько общих категорий (Event Types). Типы событий одинаковы как для стандартной, так и для распределенной систем алармов (см. табл. 3.2.).

Тип	Событие
ACK	Аларм был подтвержден
ALM	Возникла аварийная ситуация
EVT	Возникло аварийное событие
RTN	Переменная перешла из аварийного состояния в обычное
SYS	Возникло системное событие
USER	Изменение значения переменной $Operator
DDE	Получено значение переменной от DDE - клиента
LGC	Скрипт изменил значение переменной
OPR	Оператор ввел новое значение переменной

Таблица 3.2.

Первые шесть событий выбираются автоматически при разрешении регистрации событий. Для остальных трех событий разрешение регистрации устанавливается при определении переменной в словаре переменной.

3.2.2. Приоритеты алармов

Каждому аларму в InTouch соответствует некоторая величина, называемая приоритетом аларма. Этот приоритет характеризует важность данного аларма и принимает значения от 1 до 999 (наиболее серьезные алармы имеют приоритет 1). Организовав несколько диапазонов значений и связав алармы с каждым диапазоном, можно достаточно легко отфильтровать критические алармы от некритических. Выполнение анимационных функций, скриптов подтверждения, печать и просмотр информации также могут зависеть от приоритетов.

В частности, возможно следующее распределение приоритетов по четырем группам важности алармов (табл. 3.3.):

Алармы	Диапазон приоритетов
Критические	0 - 249
Существенные	250 - 499
Несущественные	500 - 749
Информационные	750 - 999

Таблица 3.3.

При определении InTouch - переменных и условий возникновения алармов каждый из них может связываться с определенным диапазоном при указании приоритета из этого диапазона. Определив уровни приоритетов, пользователь получает возможность просмотра и печати тех алармов, которые интересуют его в текущий момент.

3.2.3. Группы алармов

Каждый аларм связан с определенной логической группой алармов. Все эти группы определяются пользователем и могут быть организованы в иерархическую структуру до восьми уровней иерархии. Это позволяет сгруппировать алармы в зависимости от их организации, схемы размещения оборудования, приоритетов и любых других признаков. Группы алармов являются полезным средством фильтрации вывода информации об алармах на экран дисплея или принтер.

Каждая переменная связывается с какой-либо группой алармов. Если пользователь не определил такую группу для конкретной переменной, то она автоматически связывается с корневой группой алармов $System. С любой группой алармов можно связать как переменную, так и другую группу алармов. Взаимосвязи всех групп алармов представляются древовидной структурой, у которой в качестве корневой является группа $System. Все определяемые группы алармов автоматически становятся потомками этой группы.

Указанная иерархическая древовидная структура может иметь до восьми уровней, при этом каждая входящая в дерево группа может иметь до 16 подгрупп (рис.3.2.1).

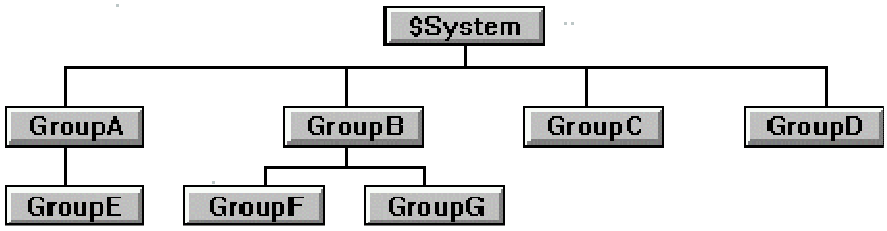


Рис.3.2.1. Иерархическая древовидная структура групп алармов.

Для создания таких групп в меню окна WindowMaker предусмотрена команда Special/Alarm Groups (группы алармов), вызывающая появление диалога Alarm Groups (рис.3.2.2). При определении переменных в словаре Tagname Dictionary нажатие кнопки Group (см. рис.2.3.4) также выводит на экран этот диалог.

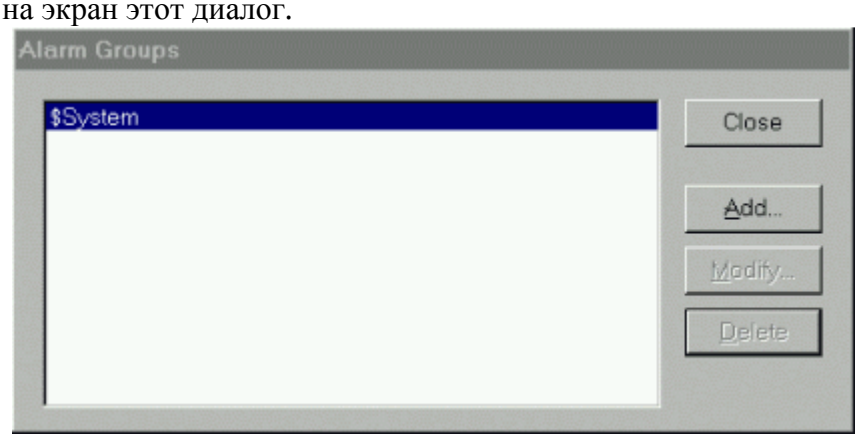


Рис.3.2.2. Диалог Alarm Group (группы алармов).

Воспользовавшись кнопкой Add, можно добавить группу алармов, а также формировать древовидную структуру системы алармов, определяя родительские группы и группы - потомки. При этом открывается диалог (рис.3.2.3) Add Alarm Group (добавить группу алармов). Кнопка Parent Group (родительская группа) предназначена для выбора родительской группы в древовидной структуре. В диалоге предусмотрено поле Comment (комментарий) для ввода необязательного текста, комментирующего данную группу.

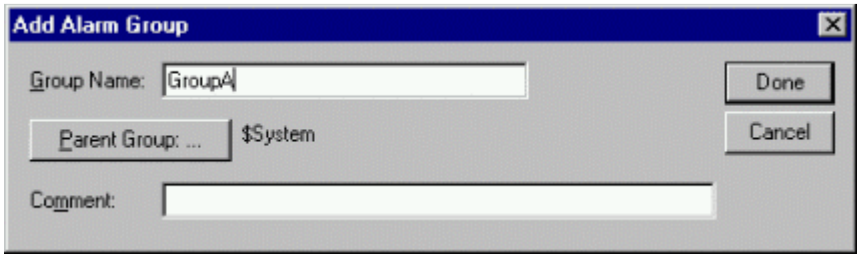


Рис.3.2.3. Диалог Add Alarm Group (добавить группу алармов).

На рис. 3.2.4 диалог Alarm Group (группы алармов) заполнен в соответствии с древовидной структурой групп алармов, представленной на рис.3.2.1.

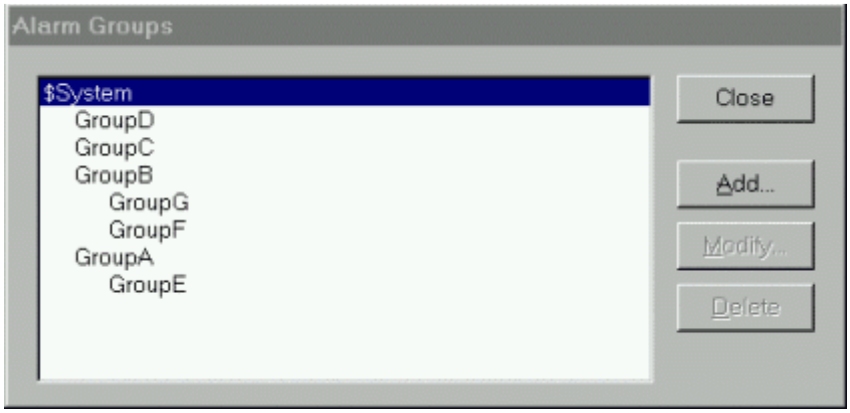


Рис.3.2.4. Диалог Alarm Group.

3.2.4. Определение условий аларма для переменной

Условия возникновения аварийных ситуаций определяются в словаре переменных (Tagname Dictionary). После выбора типа переменной откроется диалог ее подробного описания. Диалог подробного описания аналоговой переменной типа Integer I/O был приведен в предыдущем разделе (рис.2.3.7). Для дискретный переменной этот диалог имеет следующий вид:

Рис. 3.2.5. Диалог подробного описания дискретной переменной.

Поле Initial Value с опциями On-1/Off-0 (начальное значение - вкл./откл.) предназначено для задания дискретного состояния переменной в момент запуска WindowViewer (среда исполнения).

В поле Input Conversion (преобразование входных значений) указывается тип преобразования входной величины в момент обновления базы данных:

- Direct - входная величина читается без преобразования;
- Reverse - входная величина после чтения инвертируется.

Поля On Msg/Off Msg определяют текст, который будет отображен в окне вывода алармов при срабатывании аларма на ON/OFF.

3.2.5. Вывод информации об алармах

Для отображения информации об аварийных ситуациях или событиях в InTouch предусмотрены два типа объектов (окон): Alarm Summary (текущие алармы) и Alarm History (архивная сводка алармов).

С помощью объекта "Текущие алармы" на экран дисплея выводится информация только о текущих подтвержденных или неподтвержденных аварийных ситуациях. В случае возврата ситуации в нормальное состояние запись о ней исчезает из текущей аварийной сводки.

С помощью объекта "Архивная сводка алармов" на дисплей выводятся данные об аварийных ситуациях или событиях, включая количество уже произошедших аварийных ситуаций данного типа, время подтверждения, время возврата в нормальное состояние.

Создание системы алармов производится в несколько этапов:

- создание объекта (окна) вывода аварийной информации;
- конфигурирование окна вывода аварийной информации; - форматирование сообщений;
- конфигурирование системы алармов (определение общих свойств алармов, свойств регистрации и печати).

• Для создания объекта вывода алармов следует сначала вывести на экран диалоговое окно Wizard Selection (Выбор мастера). Это достигается нажатием кнопки Wizard в инструментарии InTouch.. Далее производится выбор категории Alarm Displays (окна вывода алармов) в списке мастеров, в категории выбирается стандартная система алармов (Standard Alarm Displays). Осталось щелкнуть по Ok и вставить объект вывода аварийной информации в окно (рис.3.2.6).

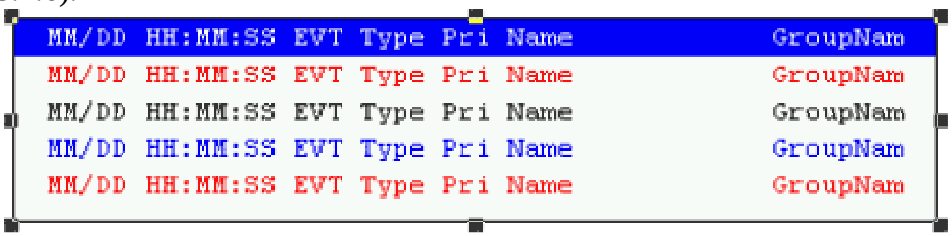


Рис.3.2.6. Стандартный объект вывода аварийной информации.

• Конфигурирование окна вывода аварийной информации производится в диалоге Alarm Configuration (параметры окна вывода аварийной информации). Вызов этого диалога производится командой Special/Animation Links меню WindowMaker (рис.3.2.7). Быстрый доступ к этому диалогу можно получить, воспользовавшись меню правой кнопки мыши с последующим щелчком на строке Properties.

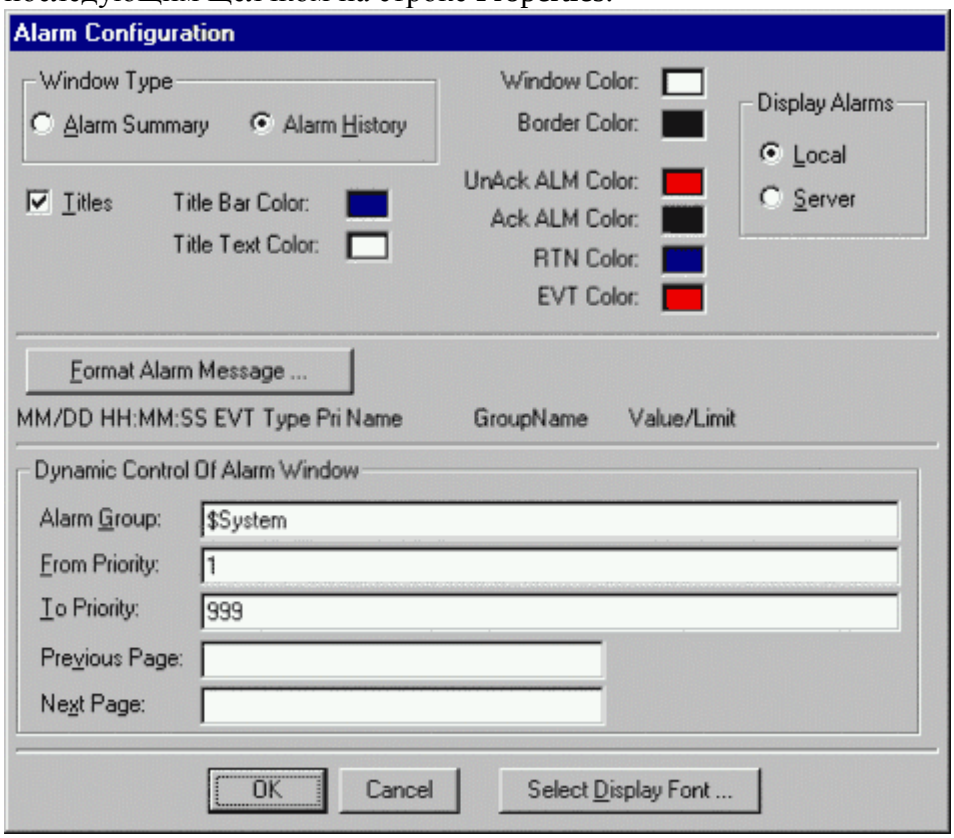


Рис.3.2.7. Диалог Alarm Configuration.

В этом диалоге определяется тип окна вывода аварийной информации ("Текущие алармы" или "Архивная сводка алармов"), группа алармов (Alarm Group), границы диапазона

приоритетов окна вывода алармов (From/To Priority), дискретные переменные для перехода на предыдущую (Previous Page) и следующую (Next Page) страницу списка алармов. Для выбора шрифтов следует воспользоваться кнопкой Select Display Font.

Нажатие кнопки Format Alarm Message (форматирование аварийного сообщения) выводит на экран одноименный диалог (рис.3.2.8), где определяется информация, включаемая в аварийное сообщение.

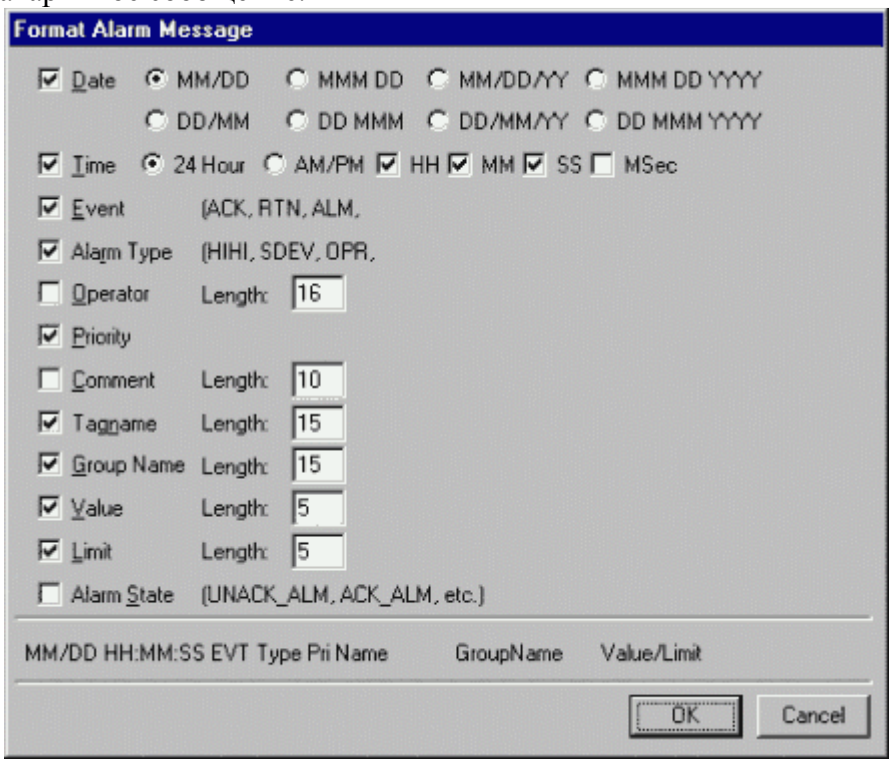


Рис.3.2.8. Диалог Format Alarm Message.

В строку аварийного сообщения можно включить текущую дату (Date), текущее время (Time), тип аларма (Alarm Type), приоритет (Priority), имя переменной (Tagname), ее текущее значение (Value), а также группу алармов (Group Name) и статус аларма (Alarm State). Пример формата строки аварийных сообщений приведен на рис.3.2.6.

3.2.6. Конфигурирование стандартной системы алармов

В соответствии с алгоритмом настройки системы алармов InTouch следующий этап предполагает настройку системы алармов в целом, т. е. определение общих свойств системы, а также свойств регистрации и печати алармов. Для входа в диалог конфигурирования стандартной системы алармов следует воспользоваться командой Special/Configure/Alarms либо в группе Configure окна Application Explorer дважды щелкнуть на строке Alarms. На экране появится диалоговое окно Alarm Properties (Свойства алармов) с открытой страницей General (Общие).

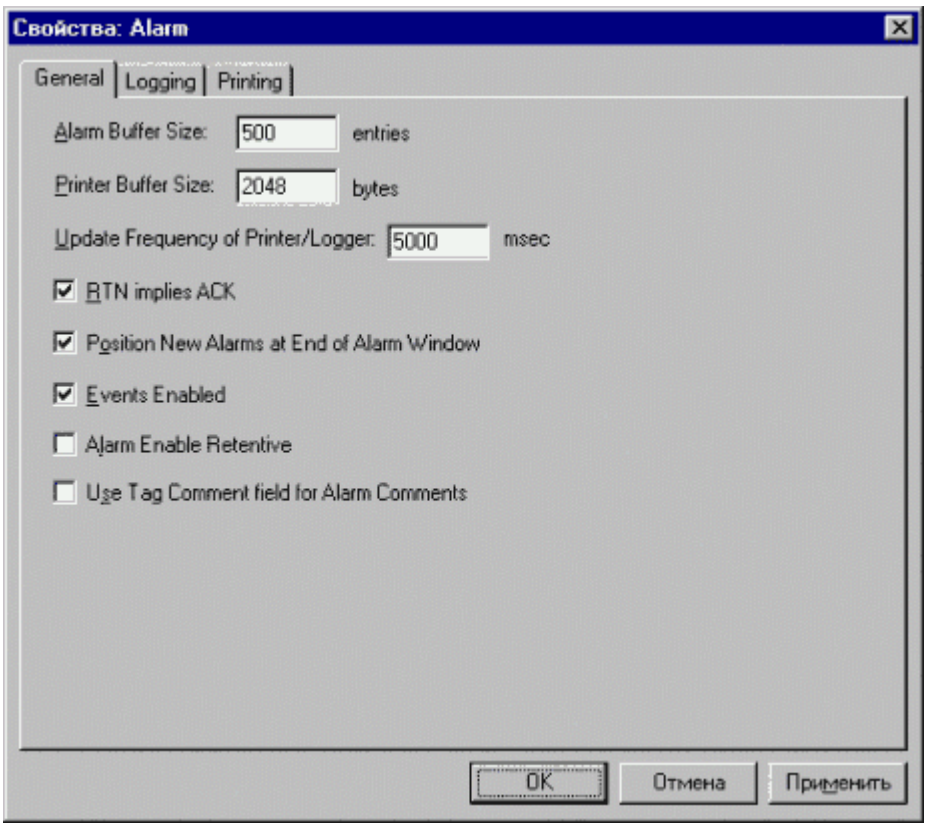


Рис.3.2.9. Диалог Alarm Properties (Свойства алармов).

Не останавливаясь подробно на описании полей этого диалога, следует отметить лишь, что пользователь может здесь определить самые различные параметры стандартной системы алармов:

- количество записей об аварийных ситуациях, которые одновременно будут находиться в буфере алармов;
- размер буфера печати подключенного к параллельному порту принтера;
- период времени в миллисекундах, через который WindowViewer будет периодически обращаться к принтеру;
- поведение окна при добавлении нового аварийного сообщения к списку;
- разрешение регистрации событий, связанных с изменением данных в результате операций ввода/вывода, действий оператора, скрипта или системы и т. д.

Кроме того, возможно определение параметров регистрации и печати событий и алармов.

Параметры регистрации алармов/событий.

Кроме возможности отображения информации об аварийных ситуациях на экране дисплея InTouch позволяет сохранять ее на жестком диске компьютера. Регистрационный файл является обычным ASCII - файлом и может впоследствии обрабатываться любым текстовым редактором. Генерация файлов, определение максимальной длины, вид регистрируемой информации, срок хранения регистрационных файлов на диске и другие параметры задаются пользователем.

Для определения параметров регистрации в файле надо щелкнуть на закладке Logging (Регистрация) диалога Alarm Properties (рис.3.2.9).

Параметры печати.

В дополнение к выводу информации об аварийных ситуациях на экран дисплея и в регистрационный файл на диск возможен вывод ее и на печать. Содержание выводимой информации определяется пользователем на закладке Printing (Печать) диалога Alarm Properties (рис.3.2.9).

Во время печати информации об аварийных ситуациях порт принтера находится под полным контролем InTouch, поэтому для этой цели необходимо использование отдельного принтера. Никакое другое приложение, включая операционную систему, не сможет пользоваться этим принтером в обычном режиме, пока не будет запрещен вывод аварийной информации.
На этой закладке можно определить следующие параметры печати:

- порт, к которому подключен принтер;
- параметры этого порта (скорость передачи, вид контроля четности, разрядность данных, количество стоповых битов);
- формат аварийных сообщений (все отображаемые, записываемые на диск и печатаемые сообщения форматируются одинаковым образом).
- группу аварийных ситуаций; - значение приоритета регистрируемой аварийной ситуации.

Работа с удаленными алармами.

Основное назначение стандартной системы - это отображение аварийных ситуаций и событий, возникающих на одном (локальном) InTouch - приложении. Вместе с тем Wonderware предлагает возможность использовать эту систему и для сетевых приложений. Но при этом должно быть выполнено следующее требование: в каждом узле должна функционировать идентичная копия InTouch - приложения и NetDDE. Одно из приложений конфигурируется как сервер алармов, который снабжает аварийной информацией один или несколько удаленных узлов. Сохраняется возможность подтверждения отдельных алармов и групп алармов.

Для создания такой конфигурации системы алармов следует при определении параметров окна вывода аварийной информации (диалог Alarm Configuration, рис.3.2.7) отметить опцию Server в поле Display Alarms для просмотра аварийной информации, накопленной узлом сервера.

На следующем этапе должно быть произведено конфигурирование сервера алармов в диалоге Свойства WindowViewer (рис.3.2.10). Этот диалог вызывается командой Special/Configure/WindowViewer. Для быстрого вывода этого диалога надо дважды щелкнуть на строке WindowViewer группы Configure окна Application Explorer.

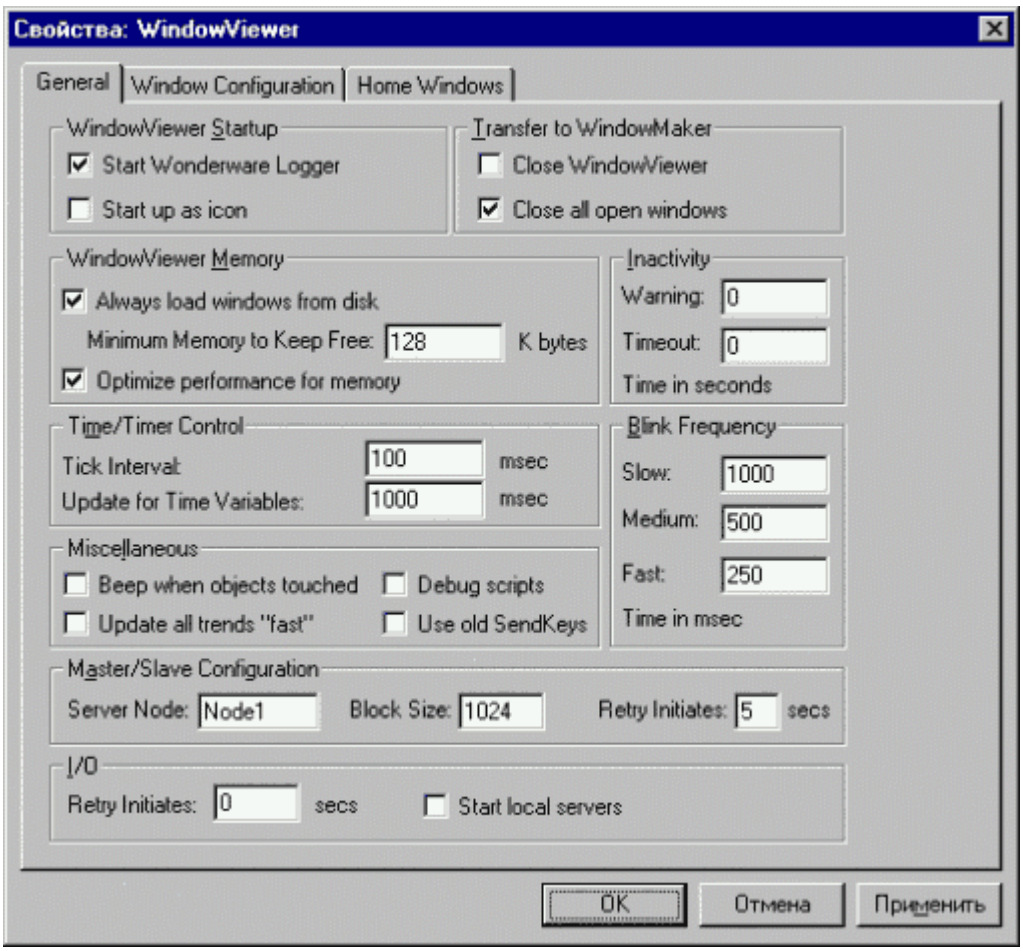


Рис.3.2.10. Диалог Свойства WindowViewer.

В группе Master/Slave Configuration (архитектура ведущий/подчиненный) в поле Server Node (имя серверного узла) следует ввести имя узла с сервером алармов, от которого удаленные узлы будут получать аварийную информацию.

3.2.7. Распределенная система алармов

Стандартную систему алармов рекомендуется использовать для идентичных InTouch - приложений. Распределенная система расширяет возможности стандартной и позволяет подтверждать аварийные ситуации, генерируемые системами алармов других включенных в сеть lnTouch-приложений.

Распределенная система имеет следующие характеристики:

- Возможность отображения и подтверждения алармов любого InTouch- узла сети.
- Новый объект вывода с линейками прокрутки, настраиваемой шириной столбцов, возможностью выбора группы алармов, панелью состояния, динамическими типами вывода и различными цветами для разных приоритетов алармов.
- Функции QuickScript, реализующие динамическое управление отображением и подтверждением алармов.
- Механизм группирования, обеспечивающий одновременное обращение к нескольким контрольным группам разных приложений по одному имени.
- Возможность добавления комментариев к аварийной информации при подтверждении алармов.

Поскольку распределенная система является расширением стандартной, то она обладает такими же параметрами, как и стандартная (уже описанными ранее).

3.3. Алармы в Citect

3.3.1. Типы алармов

Citect поддерживает два типа алармов:

- аппаратные алармы;
- конфигурируемые алармы.

Аппаратные алармы призваны информировать оператора о неисправностях, возникающих в устройствах (Hardware) системы управления (контроллерах, модулях ввода/вывода, каналах связи). Citect постоянно запускает диагностические процедуры для проверки как собственного состояния, так и состояния всего периферийного оборудования независимо от желания оператора. Сведения об обнаруженных неисправностях выводятся оператору автоматически. Это свойство Citect является встроенным и не нуждается в предварительной настройке (конфигурировании). Аппаратные алармы отображаются на специальной странице (Hardware Alarm Page).

Алармы, вызываемые отклонениями технологических параметров за допустимые границы, неисправностью технологического оборудования, надо предварительно конфигурировать. Система алармов Citect позволяет конфигурировать алармы по отдельным переменным, по группам переменных, по выражениям, по результатам расчетов и т. д.

В Citect различают четыре типа алармов:

- дискретные (digital) алармы;
- аналоговые (analog) алармы;
- алармы с метками времени (time stamped);
- составные (advanced) алармы.

Дискретные алармы срабатывают при изменении состояния дискретной переменной. При этом для срабатывания аларма можно использовать любое из двух состояний: TRUE / ON (1) или FALSE / OFF (0). По умолчанию аларм срабатывает, когда переменная принимает значение TRUE / ON (1). Если при конфигурировании аларма перед именем переменной поставить логический оператор NOT, это приведет к инвертированию логики. Аларм сработает, когда переменная примет значение FALSE / OFF (0). Например, для создания дискретного аларма, срабатывающего при выключении насоса (переменная PUMP), в поле имени переменной надо ввести NOT PUMP и аларм сработает на FALSE/OFF (0).

Citect допускает возможность конфигурирования дискретного аларма в зависимости от изменения состояния одной или двух дискретных переменных. Если определены две переменные, то они обе должны изменить свое состояние для срабатывания аларма. Для создания аларма, срабатывающего при одновременно открытых двух клапанах, достаточно в соответствующие поля ввести имена переменных, например, VALVE1 и VALVE2. Аларм сработает, когда оба клапана будут в состоянии TRUE / ON.

Аналоговые алармы базируются на анализе выхода значений переменной за указанные верхние и нижние пределы. Аналоговые алармы могут быть заданы в нескольких комбинациях (см. раздел 3.1):

- High и High High (верхний и выше верхнего);
- Low и Low Low (нижний и ниже нижнего);
- Deviation (отклонение от нормы);
- Rate of Change - ROC (скорость изменения).

Алармы с меткой времени подобны дискретным алармам - аларм срабатывает при изменении дискретного параметра. Однако эти алармы имеют точную привязку ко времени (с разрешением в 1 миллисекунду !!!), которая позволяет установить точное время его срабатывания. Таймер обычно считывает время из устройства ввода/вывода. Миллисекундная точность позволяет выявлять взаимосвязи между алармами.

Составные алармы срабатывают, когда результат выражения Cicode меняет значения от FALSE к TRUE. Они требуют большего времени на обработку, чем другие типы алармов. Поэтому большое количество составных алармов существенно ухудшает характеристики

системы управления. Составные алармы рекомендуется использовать лишь в том случае, когда невозможно применить другие типы алармов.

3.3.2. Конфигурирование алармов

Конфигурирование алармов можно производить в Citect Explorer или в Project Editor. В первом случае следует выбрать проект и открыть папку Alarms. В окне содержания проектов (Contents) появятся четыре иконки, каждая из которых предназначена для конфигурирования определенного типа алармов. В Project Editor для конфигурирования алармов потребуется открыть меню Alarms и выбрать соответствующую команду.

На рис.3.3.1 приведен интерфейс Citect Explorer с открытой папкой Alarms.

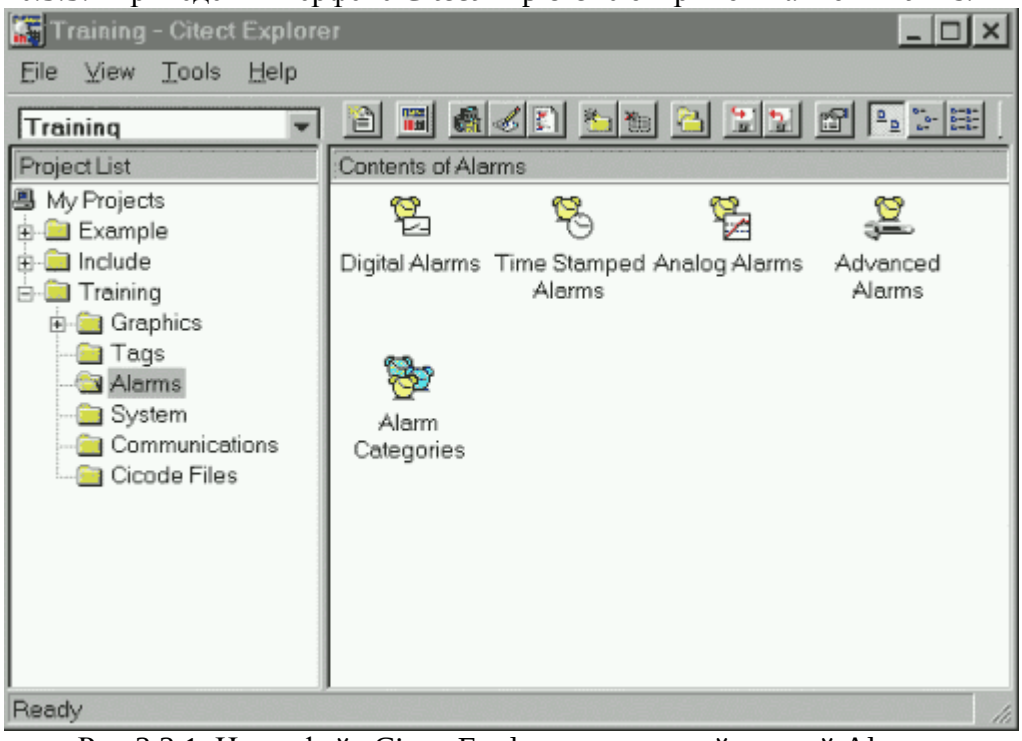


Рис.3.3.1. Интерфейс Citect Explorer с открытой папкой Alarms.

Двойной щелчок по любой из представленных в поле Contents иконок вызывает появление на экране соответствующего диалога конфигурирования аларма. На рис.3.3.2 и 3.3.3 приведены диалоги для конфигурирования дискретного и аналогового алармов.

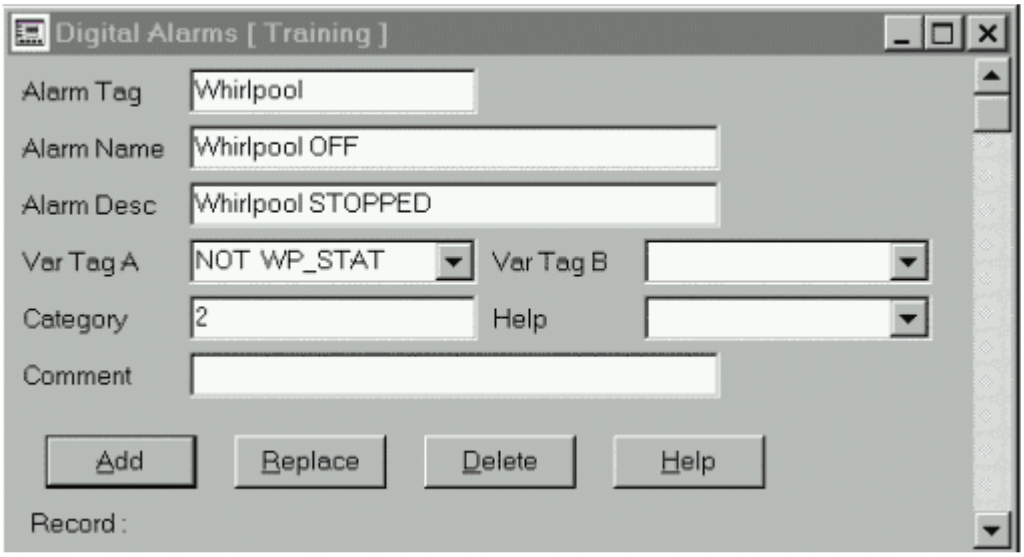


Рис.3.3.2. Диалог для конфигурирования дискретного аларма.

Читатель уже обратил внимание на поле Var Tag A. Имени переменной WP_STAT предшествует логический оператор NOT. Значит, дискретный аларм сработает на FALSE/OFF.

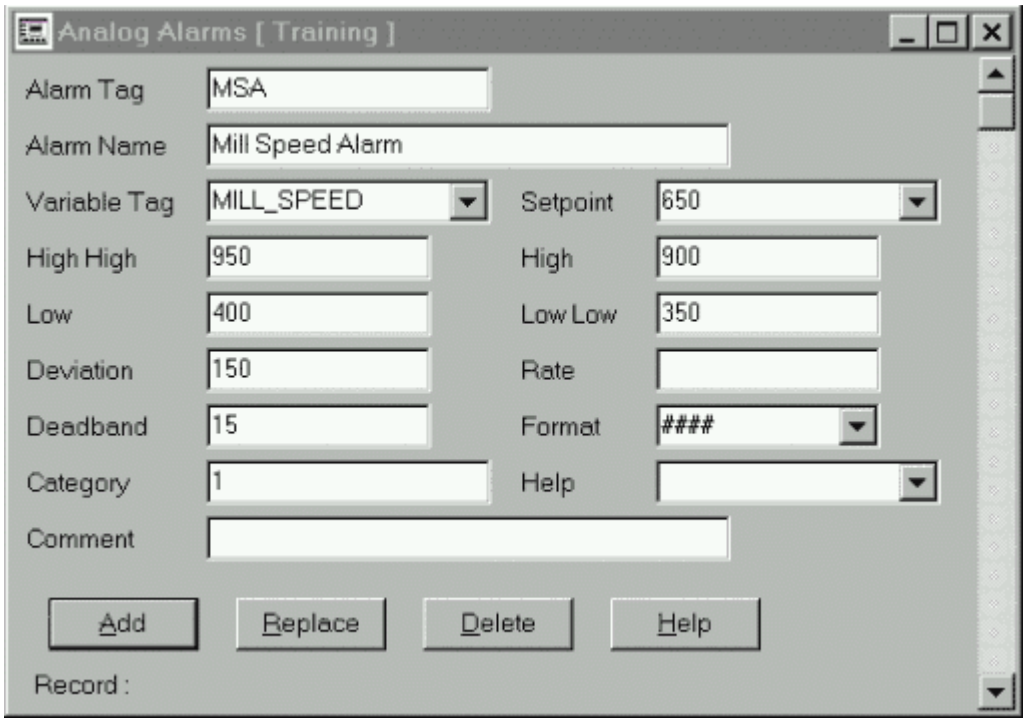


Рис.3.3.3. Диалог для конфигурирования аналогового аларма.

Каждый тип аларма имеет свои специфические параметры (поля) для настройки, но имеются и общие для всех типов алармов параметры:

- Alarm Tag - имя аларма;
- Alarm Name - имя физического устройства, связанного с алармом;
- Variable Tag - переменная, вызывающая аларм;
- Category - номер группы (категории) аларма (см. ниже).

Первые два понятия - Alarm Tag и Alarm Name - используются системой Citect только для организации вывода алармов на монитор и их регистрации (на диск, принтер и т. д.). В нижней части каждого диалога размещены четыре кнопки: Add (добавить связь), Replace (заменить), Delete (удалить), Help (справка). Конфигурирование любого аларма завершается нажатием кнопки Add. Для конфигурирования следующего аларма надо вновь заполнить поля диалога и снова нажать кнопку Add. При каждом нажатии этой кнопки срабатывает счетчик, и в поле Record появляется число, характеризующее общее количество алармов данного типа в проекте. Таким образом, при конфигурировании большого количества алармов данного типа достаточно один раз войти в соответствующий диалог и произвести конфигурирование всех алармов данного типа.

В правой части диалога имеется линейка для просмотра всех созданных алармов данного типа. Это дает возможность редактировать ранее созданные алармы. Заканчивается редактирование аларма нажатием кнопки Replace. В отличие от дискретных и аналоговых алармов составные алармы срабатывают на результат выражения Cicode (рис.3.3.4).

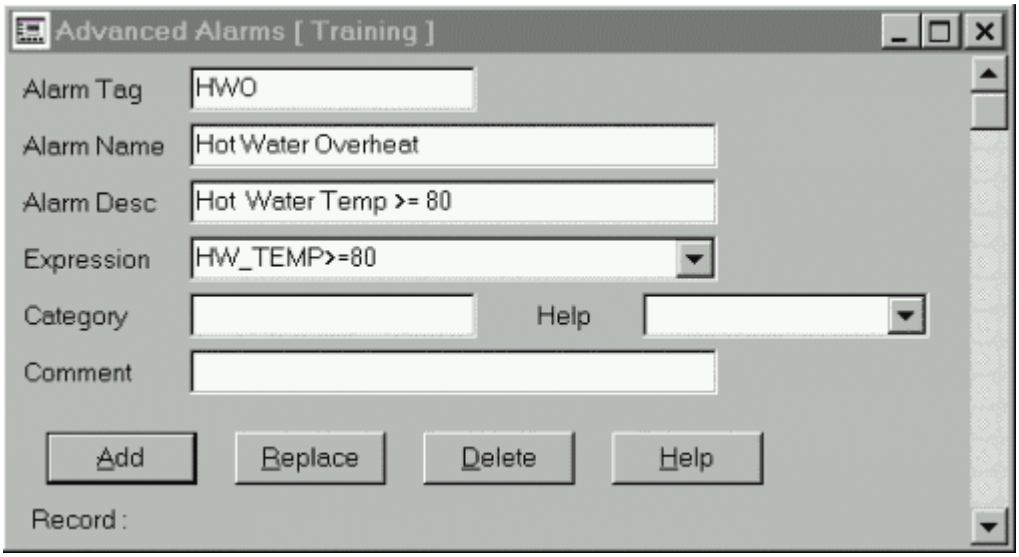


Рис. 3.3.4. Диалог Advanced Alarms.

Cicode - выражение состоит из базовых элементов языка Cicode. В этом выражении могут быть константы, значения переменных, а также результаты сложных вычислений. В рассматриваемом диалоге выражение HW_TEMP>=80 имеет следующий смысл: запустить состояние аларма, когда значение некоторой переменной HW_TEMP будет больше или равно 80 (True).

3.3.3. Категории алармов

В системе Citect предусмотрена возможность классифицировать алармы по самым различным признакам: по участкам производства, по типу алармов, имени, приоритету и т. д. В зависимости от этого каждый аларм может быть отнесен к определенной категории, и каждая категория обрабатывается как группа. Для каждой категории можно установить индивидуальные атрибуты отображения элементов аларма (шрифт и тип страницы), способ регистрации (на принтер или в файл) и действие, производимое тогда, когда срабатывает аларм определенной категории (например, включение звукового сигнала).

При разработке проекта можно определить до 255 категорий. Если категория для аларма не установлена, аларм будет иметь такие же атрибуты, как и аларм категории 0. Категория 255 используется для всех аппаратных алармов. Если не определять категорию аларма 0 или 255, Citect использует значения по умолчанию для этих категорий.

Каждая категория может иметь свой приоритет. Приоритеты алармов могут быть использованы для определения порядка их появления, обеспечивая необходимую для оператора фильтрацию. Важность приоритета уменьшается с увеличением его значения от 1 до 255. Таким образом, приоритет с номером 1 - самый высокий. Например, если алармы с приоритетами от 1 до 8 должны выводиться на экран, то первыми будут выводиться алармы с приоритетом 1 в порядке их поступления, затем - алармы с приоритетом 2 и т. д.

Задание свойств категории алармов производится в специализированном диалоге Alarm Categories, приведенном на рис. 3.3.5.

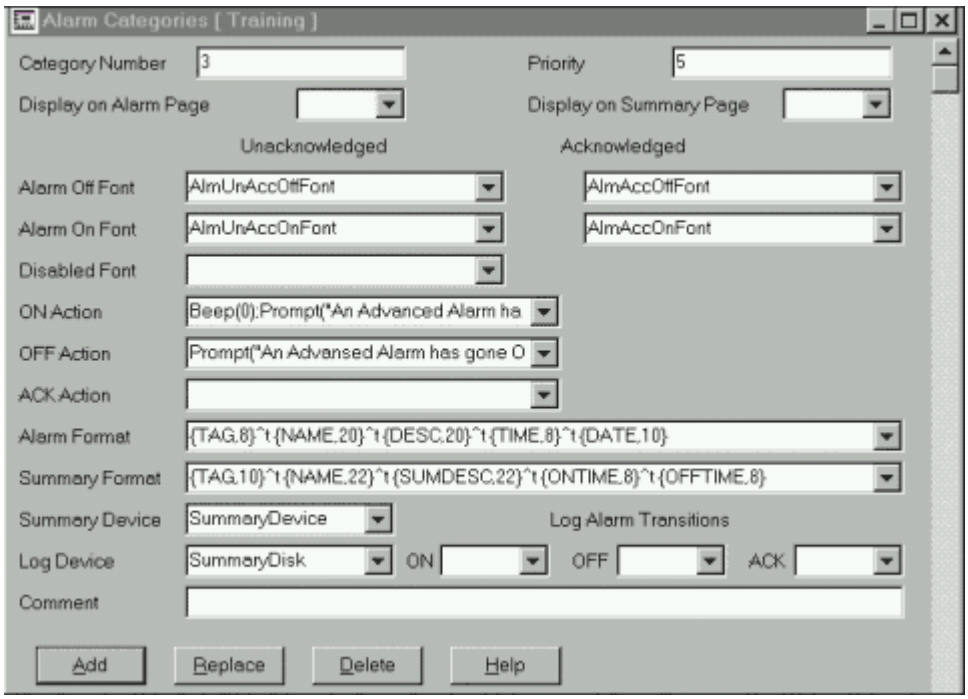


Рис. 3.3.5. Диалог Alarm Categories.

Поля Alarm On Font и Alarm Off Font предназначены для выбора шрифтов при выводе "включенных" (активных) алармов и "выключенных" алармов (переменная возвратилась в нормальное состояние).

Поля ON Action и OFF Action предписывают действие, которое должно быть реализовано при включении (выключении) аларма. Действие задается командой на языке Cicode.

Поле ACK Action предписывает действие, которое должно быть реализовано при подтверждении аларма. Также как и для предыдущих полей, действие задается командой на языке Cicode.

Каждый аларм может быть представлен на странице текущих алармов (Alarm Display) и в сводке алармов (Alarm Summary) одной строкой. Поля Alarm Format и Summary Format определяют формат вывода всех алармов данной категории на этой странице. Символ ^t между полями формата означает признак табуляции (выравнивание выводимой информации в полях формата). Действие этого формата распространяется только при отображении алармов на экран.

Поля Log Alarm Transitions (ON, OFF, ACK) определяют момент регистрации алармов данной категории (когда включается, выключается, подтверждается).

3.3.4. Отображение алармов

Для предоставления оператору информации об алармах в Citect можно создавать страницы текущих алармов (Alarm Display) и страницы сводки алармов (Alarm Summary). Более того, Citect предлагает для этих целей готовые шаблоны. Основные типы таких шаблонов приведены в главе 1. После создания новой страницы с использованием шаблона следует произвести ее конфигурирование в диалоге Properties (свойства страницы, рис.3.3.6).

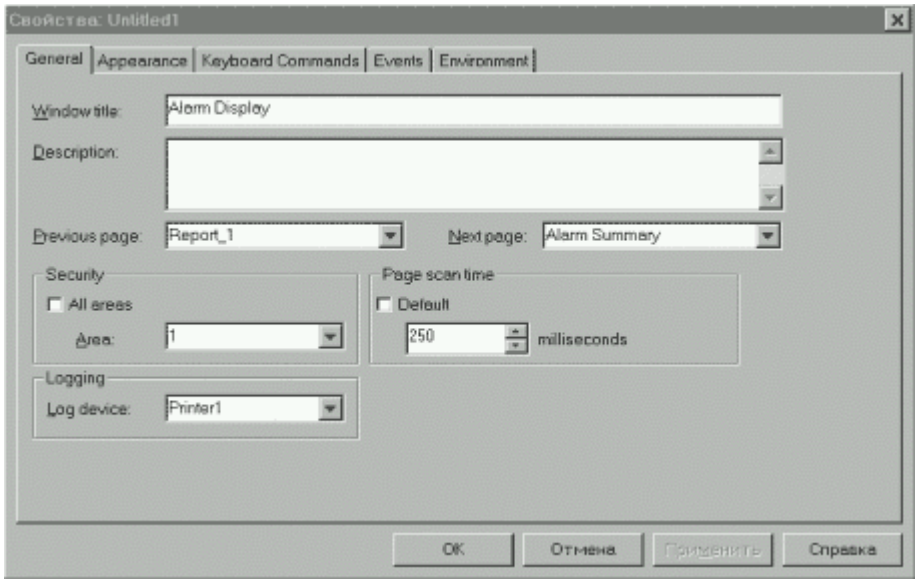


Рис.3.3.6. Диалог Properties (свойства страницы).

Этот диалог содержит несколько закладок, на которых можно определить заголовок окна в режиме исполнения (Window Title), предыдущую и последующую страницы (Previous, Next) в порядке их расположения в проекте, время обновления (scan time), видимые размеры окна, его стиль (закладка Appearance), клавиши и команды, выполняемые при их нажатии (закладка Keyboard Commands), команды, выполняемые при закрытии или открытии окна (закладка Events) и т. д.

Когда страницы для отображения алармов созданы, остается произвести конфигурирование алармов в соответствующих диалогах с присвоением категории и заполнить диалог Alarm Categories для каждой категории. При запуске режима исполнения алармы будут появляться на страницах алармов.

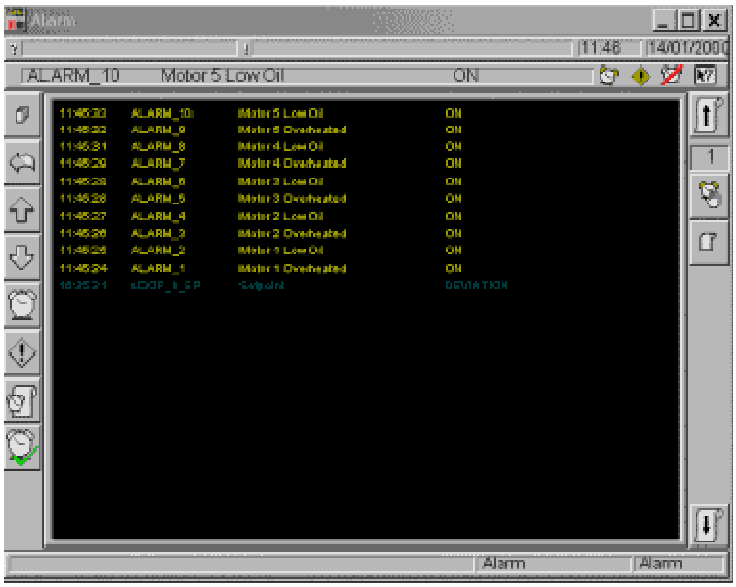


Рис. 3.3.7. Страница текущих алармов Alarm Display.

Пример страницы текущих алармов Alarm Display приведен на рис.3.3.7.
Возможные выводимые поля в Alarm Display (текущие алармы):

- имя переменной, имя аларма, описание аларма;
- категория аларма, справочная информация, зона, уровень доступа;
- тип или состояние аларма: заблокирован, подтвержден, не подтвержден;

- время/дата смены состояния или подтверждения аларма: время и дата возникновения, время и дата окончания, время и дата подтверждения, длительность.

Для дискретных алармов имеется поле состояния: on (вкл.), off (выкл.).

Для алармов с метками времени в поле времени и даты добавлена информация о миллисекундах. Для аналоговых алармов предусмотрены поля для состояний (HiHi, Hi, Lo, LoLo, Rate, Deviation), значения переменной (Value) и полосы удержания аларма (Deadband - зона нечувствительности). Так же, как и на любой графической странице, на страницах текущих алармов и сводок алармов можно расположить различные средства навигации и управления алармами (кнопки перехода на другие страницы проекта, кнопки подтверждения алармов, линейки прокрутки, регистрации алармов в файл или на принтер и т. д.). Читатель еще не забыл, что для сетевого доступа к алармам с компьютера отображения (Display Client) один из компьютеров сети должен быть сконфигурирован как сервер алармов (Alarm Server). Это может быть отдельный компьютер, играющий роль сервера алармов, либо компьютер, на который возложены функции нескольких серверов (в том числе и сервера алармов).

3.4. Подсистемы алармов в Intouch и Citect

Безусловно, основные задачи подсистемы алармов реализованы в обеих SCADA - системах. Но особенностей ее реализации достаточно много.

- Исполняющая система Citect всегда передает информацию об аппаратных (Hardware) алармах в Citect - приложениях. За разработчиком остается только решение по использованию конфигурируемых алармов. Доступность информации обо всех аварийных ситуациях в InTouch зависит от разработчика приложения.
- Подсистема алармов в InTouch и Citect является распределенной: при этом используется архитектура Client/Server. В Citect в рамках одного домена (domain) в локальной сети допустимо использование только одного сервера алармов. Остальные компьютеры могут выполнять лишь функцию клиентов по отношению к этому серверу. В InTouch допустимо произвольное количество серверов и клиентов, если брать во внимание распределенную, а не стандартную систему.
- В Citect предлагается два дополнительных типа алармов: с меткой времени и составные алармы. Последний тип алармов дает большую свободу разработчику в вопросе генерации алармов по любому условию.
- Все алармы, генерируемые приложениями в InTouch и Citect, могут быть сохранены на диске. В первом случае используются ASCII - файлы в .CSV - формате, во втором допустимыми форматами хранения являются .TXT для ASCII - файлов, а также форматы .RTF и .DBF.
- В InTouch существуют специальные графические объекты (Wizards) для отображения алармов, которые могут помещаться в любое окно (Window) приложения. При конфигурировании каждого объекта в окне определяются группы алармов с приоритетами, которые будут отображаться в объекте на этапе исполнения.
- Citect разработал шаблоны страниц (Pages), специально ориентированные на вывод как текущих и аппаратных алармов, так и сводки алармов. Компания CiT создала более высокоуровневые средства для отображения алармов. Предлагаемый инструментарий является отражением "выстраданного" опыта компании в области разработки проектов.
- Но в InTouch аналогичные решения можно получить с использованием базовых "кубиков", давая волю фантазии разработчика.

ГЛАВА 4. ТРЕНДЫ В SCADA-СИСТЕМАХ

Графическое представление значений технологических параметров во времени способствует лучшему пониманию динамики технологического процесса предприятия.

Поэтому подсистема создания трендов и хранения информации о параметрах с целью ее дальнейшего анализа и использования для управления является неотъемлемой частью любой SCADA - системы.

Тренды реального времени (Real Time) отображают динамические изменения параметра в текущем времени. При появлении нового значения параметра в окне тренда происходит прокрутка графика справа налево. Таким образом текущее значение параметра выводится всегда в правой части окна.

Тренды становятся историческими (Historical) после того, как данные будут записаны на диск и можно будет использовать режим прокрутки предыдущих значений назад с целью посмотреть прошлые значения. Отображаемые данные тренда в таком режиме будут неподвижны и будут отображаться только за определенный период.

4.1. Тренды в InTouch

InTouch предлагает пользователю оба типа графических объектов, называемых трендами: тренд реального времени и исторический (архивный) тренд. Тренды реального времени дают возможность создавать графики изменения во времени четырех переменных (4 пера), в то время как для исторических трендов можно конфигурировать до восьми перьев в одном объекте. Количество объектов типа "тренд" в приложении, в том числе и в одном окне, не ограничено.

Оба типа трендов создаются с использованием специальных графических объектов инструментальной панели WindowMaker. InTouch также обеспечивает полный контроль над конфигурированием трендов. Для примера, можно определить диапазон времени, область значений, разрешение сетки, размещение временных отметок, число перьев и атрибуты цвета и т. д. Допускается переконфигурирование архивного тренда на этапе исполнения приложения (в Runtime).

4.1.1. Архивирование (регистрация) значений переменной

При работе системы в режиме WindowViewer (среда исполнения) InTouch может производить запись значений переменных в регистрационный файл. Для того, чтобы архивирование переменной выполнялось, необходимо включить опцию Log Data (регистрация данных) при определении переменной в диалоге Tagname Dictionary (см. рис.4.1.1).

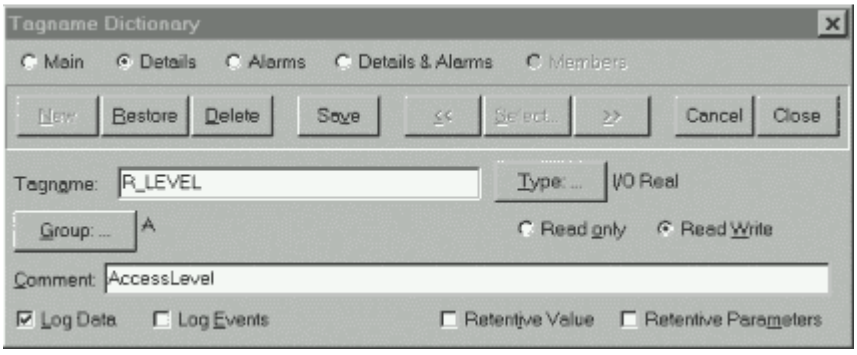


Рис.4.1.1. Диалог Tagname Dictionary с отмеченной опцией Log Data.

Запись в регистрационный файл производится всякий раз при изменении переменной на величину, превышающую порог для архивирования (Log Deadband), и по умолчанию один раз в час, если значение переменной за это время не изменилось. Поле Log Deadband находится в диалоге детального описания целой или вещественной переменной (рис.4.1.2).

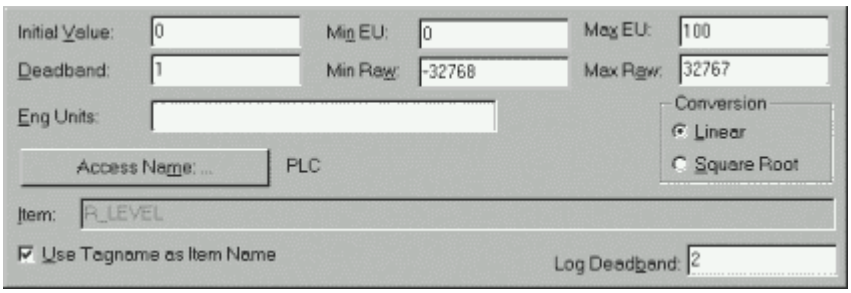


Рис.4.1.2. Диалог детального описания вещественной переменной.

Чтобы значения переменных, для которых опция Log Data разрешена, записывались в регистрационные файлы, необходимо общее разрешение глобальной функции регистрации. Его задают в диалоге Historical Logging Properties (параметры архивирования, рис. 4.1.3), который вызывается на экран командой Special/Configure/Historical Logging. В этот диалог можно также войти из окна Application Explorer.

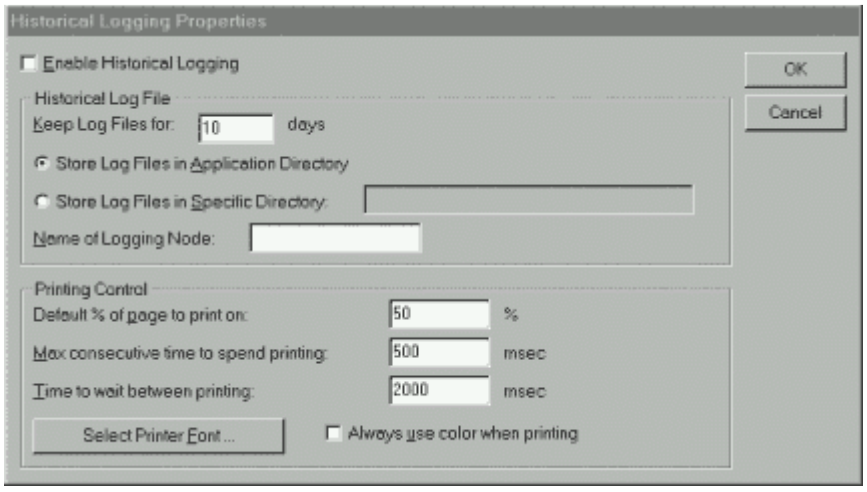


Рис.4.1.3. Диалог Historical Logging Properties.

Включение опции Enable Historical Logging дает общее разрешение на регистрацию значений переменных. Срок хранения регистрационных файлов на диске (исключая текущий день) определяется в поле Keep Log Files for в днях. Если в это поле введено значение 0, файлы будут храниться бесконечно долго. Регистрационные файлы могут быть размещены в каталоге приложения (опция по умолчанию Store Log Files in Application Directory). В противном случае следует отметить опцию Store Log Files in Specific Directory (хранить файлы в ином каталоге) и ввести полный путь до каталога, в котором будут храниться регистрационные файлы (при работе с распределенными архивами - полный сетевой путь).

Версия InTouch 7.0 (7.1) создает регистрационные файлы с расширением .LGH и .IDX. По умолчанию имена этих файлов имеют следующий формат:

YYMMDD00.LGH и YYMMDD00:IDX,

где:
- YY, MM, DD - соответственно, год, месяц и день создания файла;
- 00 - всегда нули.

Кроме того, в этом же диалоге определяются параметры печати графиков.

4.1.2. Отображение трендов

Тренды реального времени являются динамическими объектами. Они позволяют выводить изменения значений переменных, как только они происходят для любой

конкретной переменной или для выражения, которое содержит одну или несколько переменных. Данные будут появляться в окне тренда и двигаться справа налево.

Чтобы создать тренд реального времени, необходимо:

- выбрать инструмент тренд реального времени в панели инструментов WindowMaker;
- щелкнуть в окне, затем переместить мышь по диагонали и сформировать прямоугольник необходимого размера;
- отпустить кнопку мыши, что вызовет появление тренда реального времени в окне (рис.4.1.4).

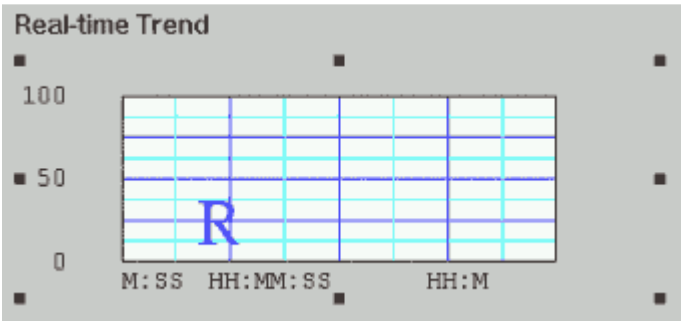


Рис.4.1.4. Объект "тренд реального времени".

При создании тренда реального времени настройки его конфигурации устанавливаются по умолчанию (настройки предыдущего тренда). Для конфигурирования тренда реального времени следует либо дважды щелкнуть на созданном объекте, либо, предварительно выбрав объект, запустить команду Special/Animation Links. На экране появится диалог Real Time Trend Configuration (конфигурирование тренда реального времени).

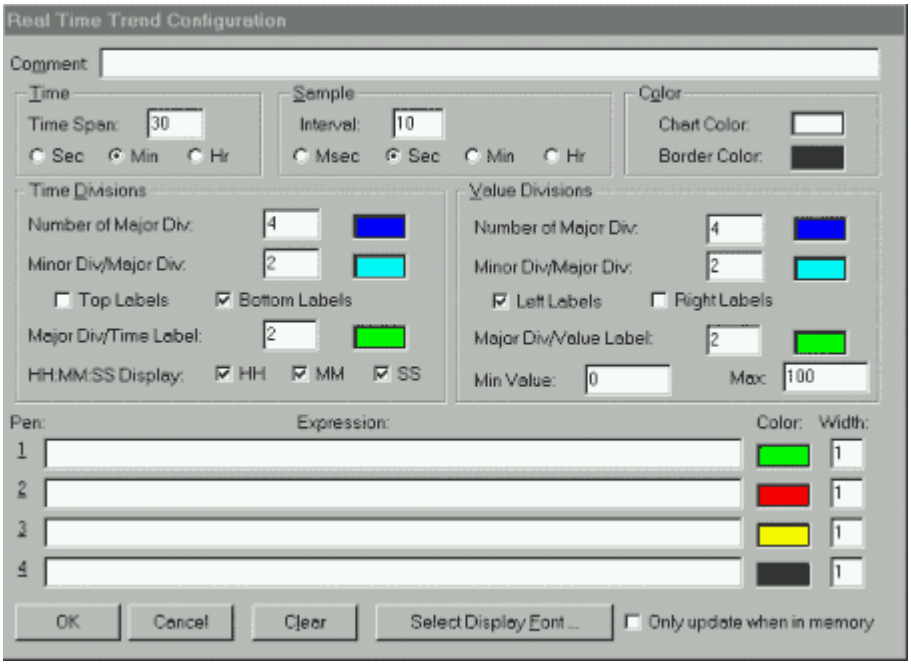


Рис.4.1.5. Диалог Real Time Trend Configuration.

Среди настроек этого диалога можно отметить диапазон времени, охватываемый трендом (Time Span), частоту вывода значение переменной (Interval), разрешение сетки по большим и малым делениям горизонтальной и вертикальной осей (Time Division, Value Division), цвета фона и рамки графика (Color). Конфигурирование перьев тренда включает выбор имени переменной или выражения, цвета и толщины линии для каждого пера (поле Expression).

Для повышения производительности системы следует отметить опцию Only update when in memory (обновлять, когда в памяти). В этом случае обновление данных тренда будет производиться только в моменты, когда окно с трендом отображается на дисплее (находится в RAM).

Есть и другие способы повышения производительности при работе с трендами реального времени (уменьшение толщины линии графика, уменьшение частоты выводы значений переменной). Например, если установлен диапазон времени (Time Span) в 30 минут, а частота вывода - 2 секунды, то число измерений, которые нужно провести за каждые 30 минут, будет равно 900 (30 * 60/2 = 900). При частоте выводе в 5 секунд число измерений существенно уменьшается: 30 * 60/5 = 360.

Исторические (архивные) тренды не являются динамическими. Они обеспечивают "снимок" состояния данных за прошедшее время, то есть по архивным данным. В отличие от трендов реального времени исторические тренды обновляются только по команде - при запуске скрипта, изменении значения выражения или нажатии оператором соответствующей кнопки.

При конфигурировании архивного тренда можно создать "визиры" (ползунки, бегунки), с помощью которых удобно получить значения всех отображаемых переменных на один и тот же момент времени. Бегунки архивного тренда представляют собой позиционные индикаторы на временной оси, положение которых определяет объем извлекаемых данных. Связав объект "движковый регулятор" с полем бегунка, можно осуществлять перемещение вдоль архивного тренда. Кроме того, имеются функции вычисления среднего, минимального и максимального значений в определенном бегунком положении. Можно создать правый и левый бегунки и производить обработку данных кривой, расположенной между бегунками. Вычисляются следующие величины: среднее, минимальное, максимальное, отношение мин/макс и стандартное отклонение. В зависимости от положения бегунков на оси можно реализовать и другие функции (увеличение и уменьшение заключенной между бегунками области графика).

Благодаря системе распределенных архивов на один и тот же график можно выводить информацию из нескольких баз данных. Все сказанное выше о механизме создания тренда реального времени инструментом Real Time Trend в среде разработки WindowMaker и о его последующем конфигурировании можно отнести и к архивному тренду, создаваемому инструментом Historical Trend среды разработки. Предлагаемый ниже способ создания и конфигурирования архивного тренда предполагает использование мастер-средств библиотеки Wizard.

Нажатие кнопки выбора мастер-средств в панели инструментов вызывает появление на экране диалога Wizard Selection (выбор мастер-средств). После выбора в списке категории Trends этот диалог будет иметь следующий вид (рис.4.1.6).

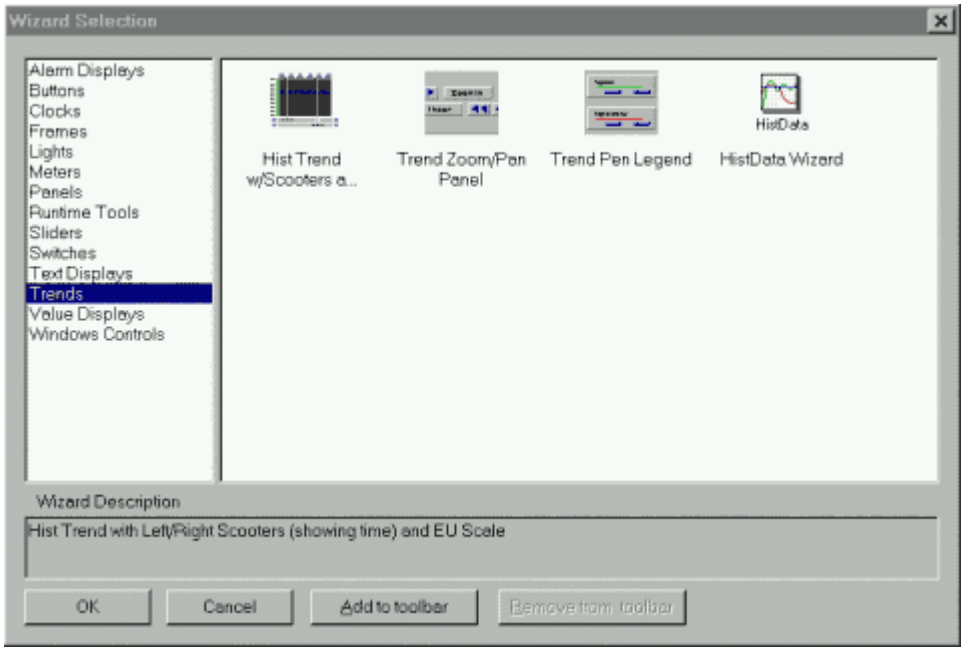


Рис.4.1.6. Диалог Wizard Selection (выбор мастер-средств).

После выбора из предложенного набора мастер-средств Hist Trend with Scooters (архивный тренд с бегунками) и щелчка по Ok программа возвращает пользователя в среду разработки. Курсор мыши при этом примет форму вставки . Последующий щелчок мыши на предполагаемом месте нахождения создаваемого объекта выводит на экран архивный тренд (рис.4.1.7). Объекты этого типа ведут себя аналогично любым другим объектам, то есть их можно перемещать, масштабировать и т. д.

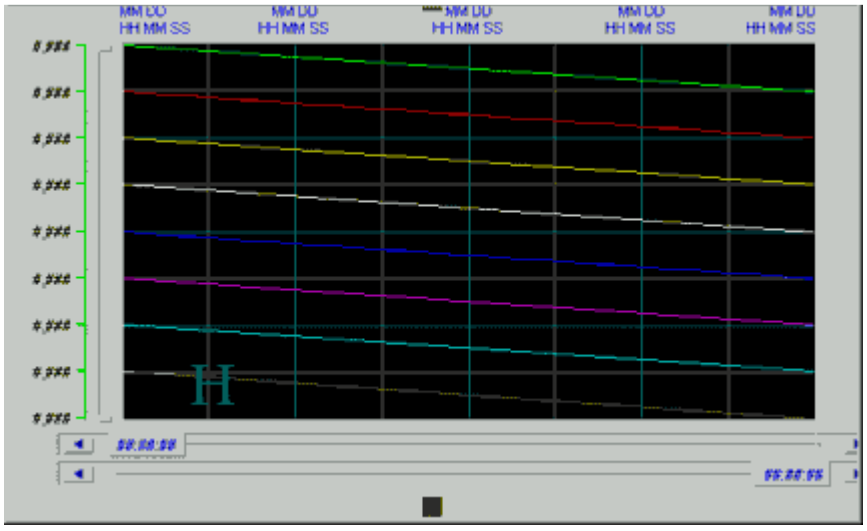


Рис.4.1.7. Объект "архивный тренд".

Двойной щелчок на объекте приводит к появлению на экране диалога конфигурирования архивного тренда (Historical Trend Char Window).

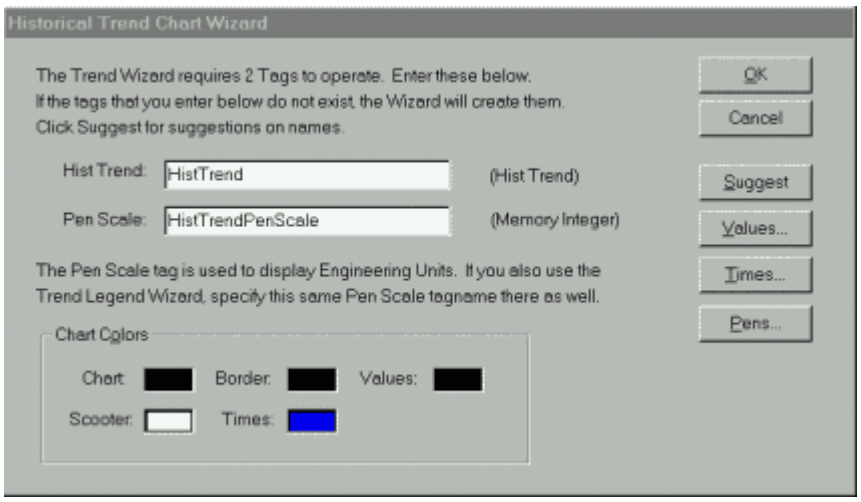


Рис.4.1.8. Диалог конфигурирования архивного тренда.

Для конфигурирования тренда с параметрами по умолчанию следует нажать кнопку Suggest (вариант). Нажатие кнопок Times и Values выводит на экран окна конфигурирования разрешения сетки по большим и малым делениям горизонтальной и вертикальной осей, цвета фона и рамки графика, временного диапазона и т. д. Кнопка Pens (перья) предназначена для настройки перьев архивного тренда.

Чтобы добавить в тренд функции масштабирования и перемещения или элементы управления перьями, следует использовать панели Zoom/Pan и Trend Pen Legend (рис.4.1.6), соответственно. Для того, чтобы эти компоненты работали совместно, они должны иметь одинаковые имена (Hist Trend).

4.1.3. Изменение параметров архивных трендов

При управлении в режиме реального времени оператор анализирует архивную информацию. Объем информации, ее временные диапазоны, объем статистических данных, необходимые для принятия решения по управлению технологическим процессом, заранее не известны. Поэтому оператор должен иметь возможность менять настройки архивных трендов, не выходя из режима Runtime. В InTouch такая возможность существует.

Для этого следует включить опцию Allow runtime changes (разрешить изменения во время исполнения) в диалоге конфигурирования архивного тренда (в книге не показан).

Теперь в режиме WindowViewer щелчок на архивном тренде будет вызывать на экран диалог изменения параметров архивного тренда (Historical Trend Setup, рис.4.1.9). В этом диалоге можно определить дату и время начала архивного тренда (поле Chart Start), его временной диапазон (Chart Length), присвоить перьям цвет и имена переменных, выбирая их из словаря.

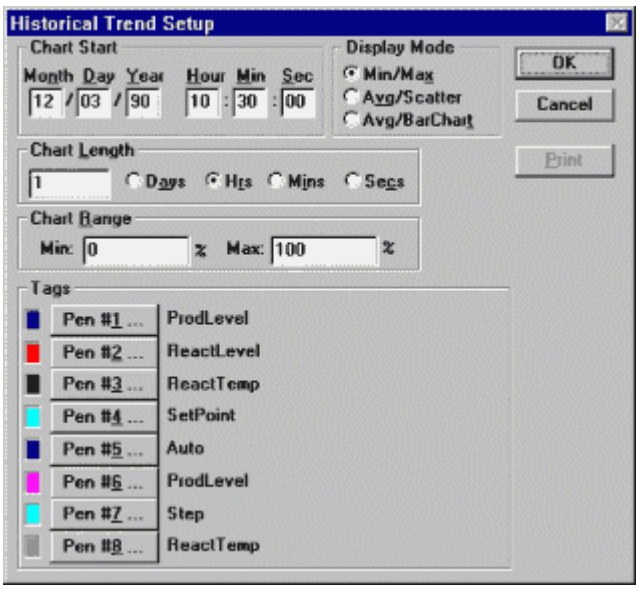


Рис.4.1.9. Диалог изменения параметров архивного тренда.

Архивный тренд может выводиться в одном из трех возможных режимах:

- Min/Max - график изменения значений переменной в виде вертикальных линий в процентах от всего диапазона, позволяющий оценить скорость изменения переменной;
- Average/Scatter - график среднего значения переменной;
- Average/Bar Chart - график среднего значения переменной в виде гистограммы.

Выбор режима производится в поле Display Mode.

4.1.4. Система распределенных архивов

В InTouch имеется система распределенных архивов, обеспечивающая поиск архивных данных в любом InTouch - приложении. Данная система расширяет возможности стандартных архивов InTouch, позволяя одновременно получать информацию из нескольких удаленных баз данных, которые в этом случае называются провайдерами архивов.

Одновременно можно обращаться к восьми провайдерам (по одному на каждое перо). Каждый узел, выполняющий функцию регистрации, может писать только в один архив.

Система, приведенная на рис.4.1.10, имеет два провайдера архивов. Левый провайдер регистрирует информацию только из узла, расположенного слева внизу. Правый провайдер регистрирует информацию из узла, расположенного справа вверху. Остальные три узла (вверху слева) лишь используют архивные данные. Читать информацию из архивных файлов может каждый из узлов системы.

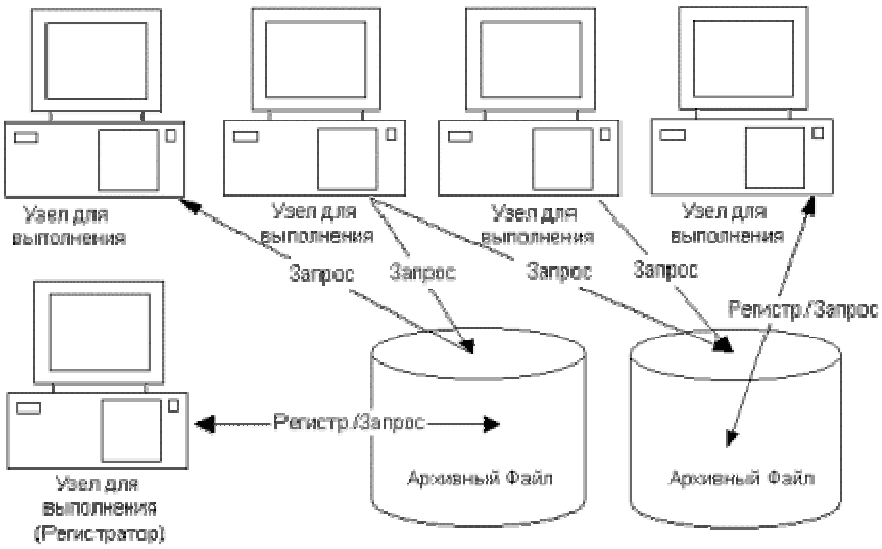


Рис. 4.1.10. Распределенная система архивов.

Создание такой системы предполагает следующие действия:

- создание списка провайдеров архивов;
- создание и определение параметров объекта "архивный тренд";
- конфигурирование приложения на удаленное архивирование данных;
- копирование приложения на все узлы.

4.2. Тренды в Citect

В системе Citect реализована единая распределенная система построения трендов реального времени и графиков для анализа технологических процессов, функционирующая в архитектуре клиент - сервер. Сбор, хранение и обработку информации для ее представления в графическом виде осуществляет сервер трендов (Trends Server). При необходимости вывода трендов реального времени и архивных трендов на экран компьютера визуализации (Display Client) клиент запрашивает у сервера необходимые данные. Таким образом, по сети передаются только пакеты "полезных данных" меньшего размера, что существенно уменьшает нагрузку на сеть.

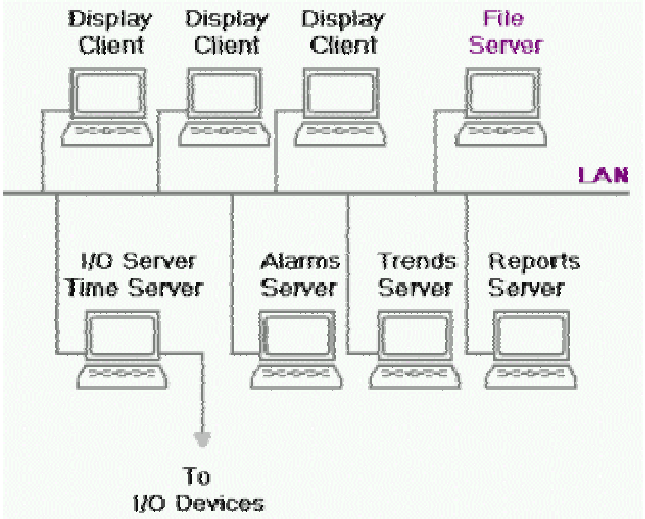


Рис. 4.2.1. Вариант сетевой архитектуры системы Citect.

Citect предоставляет возможность вывести на тренд любую переменную или значение выражения на языке Cicode. Можно одновременно посмотреть на экране любое количество трендов и увидеть до восьми графиков в любом окне тренда. Накопление данных

продолжается даже тогда, когда дисплей не активен. Можно перемещаться по страницам проекта, не влияя на процесс построения трендов и систему регистрации данных.

В Citect можно строить периодические тренды - Trend Periodic (регистрация данных через определенные интервалы времени с разрешением до нескольких миллисекунд), тренды по событию - Trend Event (регистрация данных в момент наступления события) и периодические тренды по событию -Trend Periodic Event.

4.2.1. Регистрация данных

Citect может хранить любое количество данных. Объем хранимой информации определяется размерами жесткого диска компьютера. При этом применяется эффективный метод хранения информации, максимизирующий использование дискового пространства (компрессия файлов).

Объем выборки для хранения в файлах задается в процессе конфигурирования тренда временным периодом от 10 миллисекунд до 24 часов в сутки. Конфигурирование трендов можно производить в Citect Explorer или в Project Editor. В этом случае в Citect Explorer должна быть открыта папка Tags, а в Project Editor - меню Tags (см. рис.3.3.1).

По аналогии с алармами при конфигурировании трендов используется понятие Tag (Trend Tags). Tags (теги) - это внутренние переменные системы Citect, которым присваиваются имена с целью идентификации трендовых переменных (в предыдущем разделе - алармов) при выводе их на экран и регистрации в файлы. Щелчок по иконке Trend Tags в окне Contents интерфейса Citect Explorer выводит на экран диалог конфигурирования трендов (рис. 4.2.2).

Поле Expression предназначено для ввода выражения или имени переменной, которая будет отображаться трендом. Частота выборки данных (Sample Period) вводится в формате HH:MM:SS. Можно ввести одну цифру, например 2, и это будет означать 2 секунды. Ввод десятичной цифры система воспринимает, как долю секунды. Например, 0.2 будет означать 200 миллисекунд.

Поле Type предназначено для выбора типа тренда (периодический, по событию, периодический по событию).

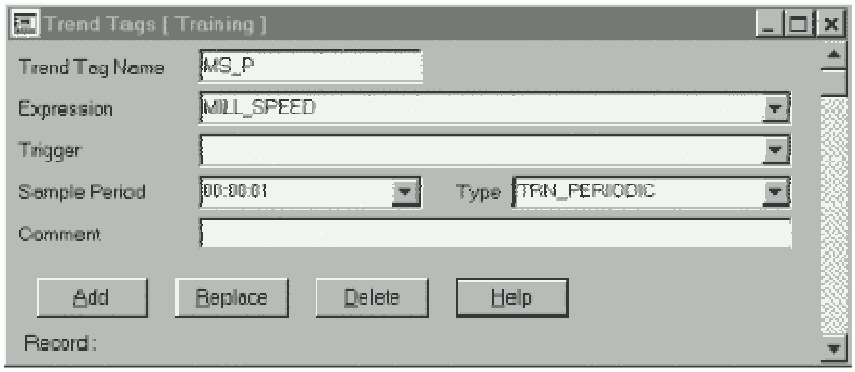


Рис. 4.2.2. Диалог конфигурирования трендов.

В нижней части диалога размещены четыре кнопки: Add (добавить связь), Replace (заменить), Delete (удалить), Help (справка). Конфигурирование тренда завершается нажатием кнопки Add. Для конфигурирования следующего тренда надо вновь заполнить поля диалога и снова нажать кнопку Add. При каждом нажатии этой кнопки срабатывает счетчик, и в поле Record появляется число, характеризующее общее количество трендов в проекте. Редактирование параметров ранее сконфигурированных трендов завершается нажатием клавиши Replace.

Считанная с устройств ввода/вывода информация используется для построения архивных трендов и сохраняется в файлы для дальнейшего анализа.

Citect использует круговую систему записи в файлы, что предпочтительней, чем в один большой файл. По умолчанию используются 10 файлов, регистрирующих данные в течение одной недели, начиная с полуночи воскресенья (рис.4.2.3). В самом начале регистрации данные записываются в первый файл. С полуночи следующего воскресенья запись будет производиться во второй файл. С полуночи следующего воскресенья запись будет производится в третий файл и т. д. После 10 недель в первый файл записываются новые данные, уничтожая при этом старую информацию. Также по умолчанию имя файла будет содержать 8 символов имени переменной тренда.

Рис. 4.2.3. Круговая система записи данных в файлы.

Частоту записи в журнал и количество используемых журнальных файлов можно изменять. Для настройки параметров файлов следует открыть диалог Trend Tags и нажать F2 для отображения дополнительных опций (см. рис.4.2.4).

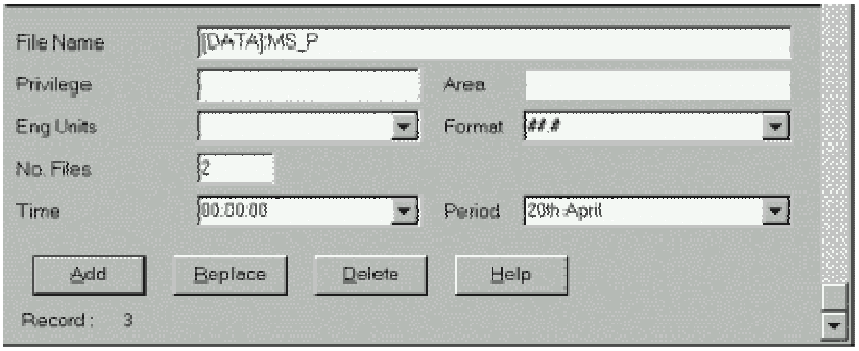


Рис. 4.2.4. Диалог настройки параметров регистрации данных в файл.

Не вдаваясь в подробное описание полей диалога, следует отметить лишь поле Format для выбора формата данных при их записи в файл (данные в файл записываются в заданном формате через запятую) и поля Time и Period для выбора временного диапазона и периода записи данных в файл. Например, если в поле Period выбрать [1 : 00 : 00], то это будет означать смену файла для записи данных каждый час. Запись 20 th Avril означает смену файла один раз в год - 20 апреля.

Пример расчета дискового пространства, необходимого для файлов тренда.
Каждое значение требует для хранения два байта. Можно предварительно рассчитать объем памяти, занимаемый архивом при его записи на диск, по следующей формуле:

$$V=464 * N +176 + (T * N * 2) / t ,$$

где:
V - объем памяти (байт);
N - количество файлов;
T - время хранения информации (сек);
t - период выборки (сек).

Например, если в архив записывается одно значение переменной каждые десять секунд в течение одной недели, и используется пять файлов данных (пять недель), то требуемый объем памяти будет равен 607296. V= 464*5 +176 + {7*24*60*60*5*2}/10=607296

4.2.2. Отображение трендов

Для отображения трендов на экране в системе Citect предусмотрены специальные шаблоны страниц:

- одиночный тренд (SingleTrend) - шаблон для создания страницы с одним окном трендов, в котором имеется до 8 перьев;
- двойной тренд (DoubleTrend-) - шаблон для создания страницы с двумя окнами трендов, в каждом из которых имеется до 8 перьев;
- сравнительный тренд (CompareTrend) - шаблон для создания страницы с двумя трендами, наложенными один на другой в целях их сравнения (до четырех пар графиков);
- масштабный тренд (ZoomTrend) - шаблон страницы с функцией масштабирования;
- выпадающий тренд (PopTrend) - шаблон для вывода тренда в любом месте экрана (в отдельном окне).
- тренды по событию (EventTrend) - шаблон страницы с одним окном для тренда по событию во времени на восемь перьев;

Эти шаблоны практически исчерпывают все потребности разработчика при создании трендов проекта. Если все-таки появится необходимость в создании нового шаблона, Citect и в этом случае предоставит свой инструмент. В графическом редакторе Graphics Builder на линейке инструментов имеется иконка NEW, щелчок по которой выводит на экран меню, одна из опций которого предназначена для создания нового шаблона (рис.4.2.5).

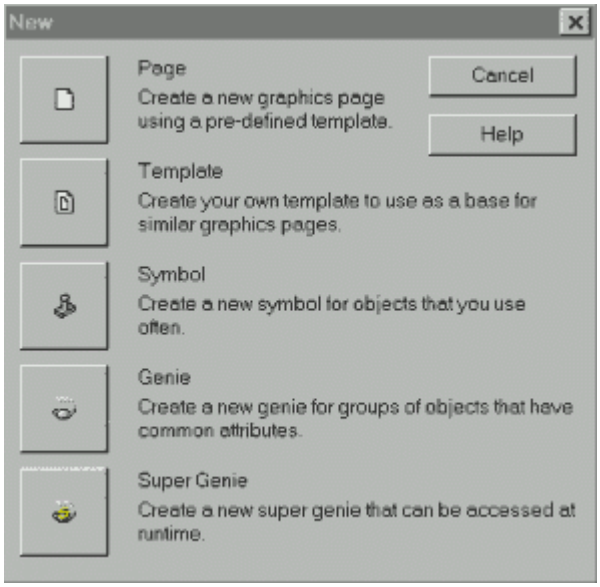


Рис.4.2.5. Меню New для создания новой страницы, шаблона, символа, джинна и суперджина.

Создание нового шаблона - интересная, творческая работа. Но читателю не менее интересно оценить то, что уже создал Citect. Тренды, созданные с помощью этих шаблонов, является

одновременно и трендами реального времени (текущие данные появляются в реальном времени в правой части графика), и архивными трендами.
Все шаблоны страниц уже снабжены различными средствами навигации и чтения значений параметров. Здесь присутствуют:

- кнопки перемещения маркера по графикам влево и вправо, при этом перемещать репер можно маленькими или большими шагами, а также в начало или конец графика;
- кнопка вывода статистических параметров - минимума, максимума, статистического среднего и стандартного отклонения;
- кнопка увеличения выделенного участка графика;
- кнопки изменения разрешения по времени и охватываемому периоду;
- кнопка, позволяющая в реальном времени менять параметры перьев;
- кнопки вывода данных графика на печать и записи в файл;
- кнопка копирования данных в буфер обмена Windows для их использования в других приложениях (в табличном формате) типа Word, Excel и т. д.

В качестве примера такого шаблона предлагается одиночный тренд (SingleTrend)-, приведенный на рис. 4.2.6.

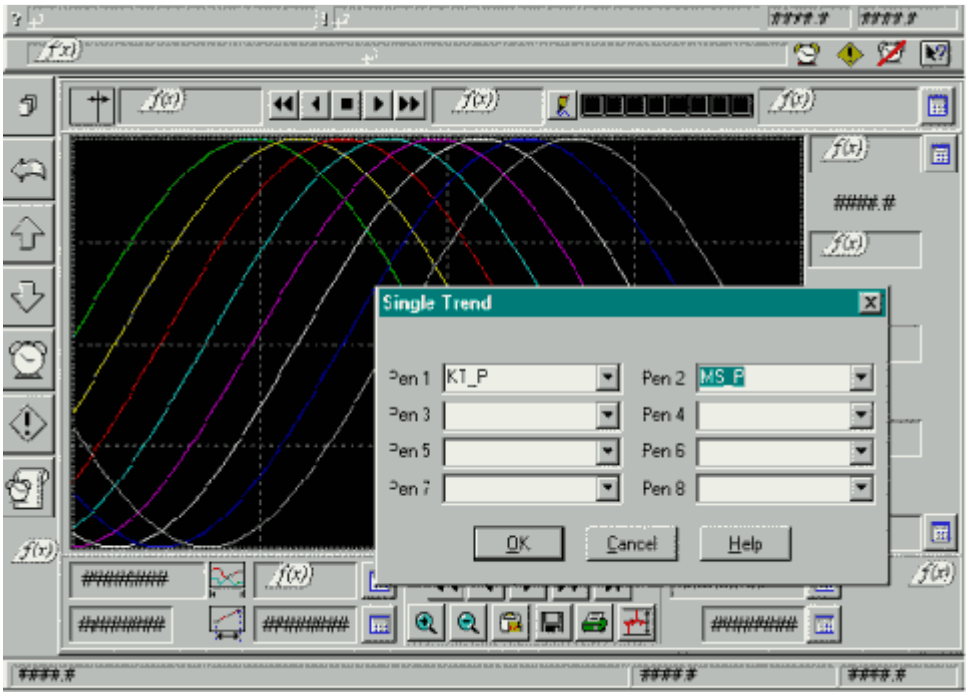


Рис.4.2.6. Шаблон одиночного тренда с окном настройки перьев.

Панель сравнения графиков предоставляет оператору возможность одновременно выводить два графика, назначив каждому перу свои временные характеристики.

Двойной щелчок мышью по полю тренда выводит на экран диалог конфигурирования перьев (8 перьев) тренда. Вводить с клавиатуры имена переменных нет необходимости. Достаточно открыть в поле каждого пера список переменных проекта и выбрать переменную, которая будет отображаться этим пером на тренде.

Для переконфигурирования перьев тренда в режиме Runtime Citect предлагает использовать специальные страницы трендов и функцию PageTrend(), позволяющую подключать к этим страницам требуемые переменные (перья). С помощью этой функции можно выводить на одну страницу тренда переменные, имеющие одну и ту же частоту выборки (одновременно не более восьми).

При создании такой страницы тренда следует все поля диалога конфигурирования перьев оставить пустыми, а функцию PageTrend( ) связать с одной из кнопок страницы меню. Теперь нажатие этой кнопки в режиме исполнения будет вызывать функцию PageTrend(sPage, sTag1 ... sTag8):

- sPage - имя страницы тренда;
- sTag1 ... sTag8 - имена переменных.

Остается ввести имя страницы тренда и имена переменных для соответствующих перьев. Например, функция PageTrend("MyTrend", "PV1", "PV2", " PV3") обеспечит вывод переменных PV1, PV2, PV3 на страницу тренда с именем MyTrend.

Все вышеизложенное делает механизм трендов в Citect удобным не только при конфигурировании (разработке), но и в процессе эксплуатации (Runtime).

При запуске режима Runtime страница одиночного тренда будет выглядеть следующим образом (рис.4.2.7).

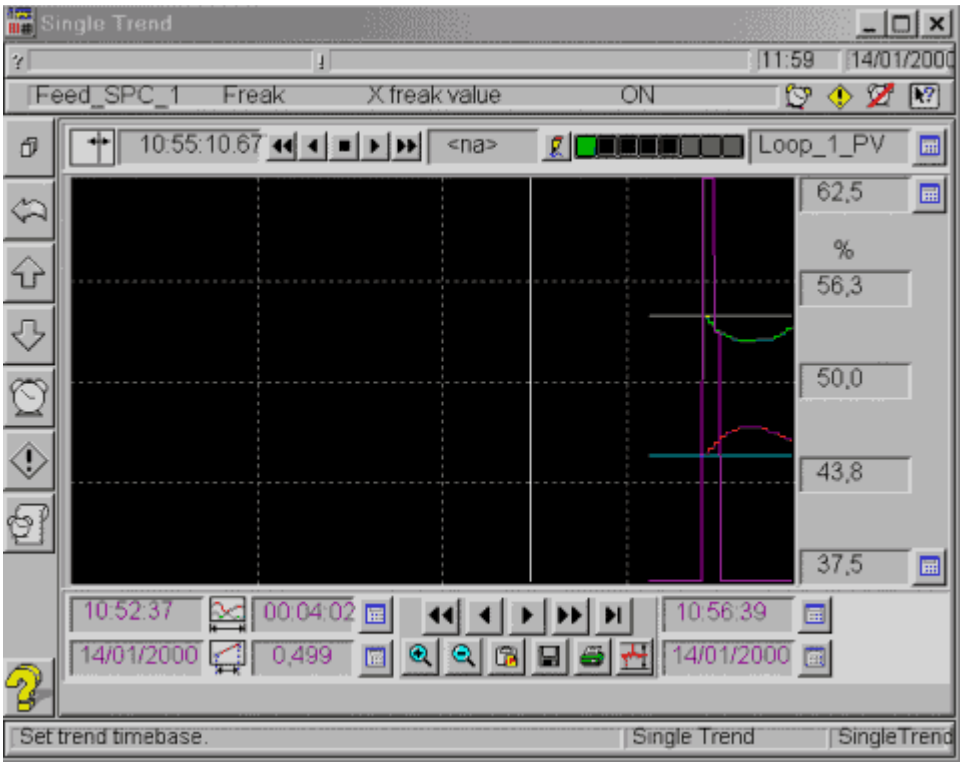


Рис.4.2.7. Страница одиночного тренда в режиме Runtime.

4.3. Отличия подсистем отображения и архивирования в InTouch и Citect

Среди программных продуктов рассматриваемого класса (SCADA - системы) InTouch и Citect выделяются своими подсистемами архивирования и отображения регистрируемых на жесткий диск данных технологического процесса.

Подсистема архивирования.

И в Citect, и в InTouch используется кольцевой буфер, принцип построения которого предполагает автоматическое уничтожение файла с самыми старыми в настоящий момент данными. В Citect непосредственно указывается количество файлов в буфере и продолжительность регистрации в файл, в InTouch - количество дней в буфере, но исполняющая система InTouch ежедневно создает новый архивный файл. (или вместо подчеркнутых слов - количество дней в буфере, а фактически количество файлов, так как исполняющая система InTouch ежедневно создает новый архивный файл.) Данные для архивирования в InTouch задаются на этапе определения переменной приложения. Таким образом, любая переменная, независимо от ее типа, может быть включена в список для архивирования (опция Log Data).

В Citect предлагается вводить дополнительную переменную (переменную типа Trend) и с ней связывать переменную Citect - приложения или выражение.

Основным режимом регистрации в InTouch является режим по изменению значения, т. е. регистрация переменной производится только в тот момент, когда изменение значения переменной превысило величину, указанную в Log Deadband.

В Citect основной режим регистрации - периодический (через определенные интервалы времени) либо по событию. Архивные файлы обеих подсистем имеют скрытый формат, что логично, но предоставляются утилиты для доступа к ним.

Подсистема отображения.

Для графического отображения архивной информации в InTouch используются два стандартных объекта (Real Time Trend, Historical Trend) и Wizard - объект, детально описанные в настоящей главе. Особенность данных объектов в том, что они могут вставляться в окно, и в них может выводиться до четырех (тренд реального времени) и восьми переменных (архивный тренд). Входящий в базовую поставку комплект Productivity Pack включает 16 Pen Trend, позволяющий выводить до 16 переменных или выражений. Каждый из указанных объектов масштабируется и поэтому может быть размещен в части окна или на всем окне.

Средства отображения архивных данных в InTouch отличаются простотой встраивания в приложение и связывания с переменными.

В Citect предлагается большое количество шаблонов различных типов трендов: один тренд (до 8 перьев) на странице, два тренда на странице и т. д. Это говорит о том, что Ci Technologies считает подобные шаблоны типичными для использования в проектах (свой опыт компания выразила в шаблонах) и предлагает их пользователям Citect.

В InTouch предлагаются стандартные объекты тренда реального времени и архивного тренда. Исторически сложилось так, что эти объекты разделены. Деление это условное хотя бы потому, что 16 Pen Trend из Productivity Pack функционирует в режиме и тренда реального времени, и архивного тренда.

Шаблоны подсистемы отображения Citect едины для трендов реального времени и архивных трендов. Они позволяют выводить на страницы трендов архивные данные для их последующего анализа, отображая в то же самое время значения переменных в реальном времени.

ГЛАВА 5. ВСТРОЕННЫЕ ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Встроенные языки программирования - мощное средство SCADA - систем, предоставляющее разработчику гибкий инструмент для разработки сложных приложений. Первые версии SCADA - систем либо не имели подобных языков, либо эти языки реализовывали небогатый набор функций. В современных версиях SCADA - систем функциональные возможности языков становятся существенно богаче. Явно выделяются два подхода:

- Ориентация встроенных языков программирования на технологов. Функции в таких языках являются высокоуровневыми, не требующими профессиональных навыков программирования при их использовании. Количество таких функций в базовых поставках не исчисляется сотнями, хотя существуют свободно распространяемые библиотеки дополнительных функций.
- Ориентация на системного интегратора. В этом случае в качестве языков чаще всего используются VBasic - подобные языки.

В каждом языке допускается расширение набора функций. В языках, ориентированных на технологов, это расширение достигается с помощью дополнительных инструментальных средств (Toolkits). Разработка дополнительных функций выполняется обычно программистами - профессионалами.

Разработка новых функций при втором подходе выполняется обычно разработчиками приложений (как и в традиционных языках программирования).

Полнота использования возможностей встроенных языков (особенно при втором подходе) требует соответствующего уровня квалификации разработчика, если, конечно, в этом есть необходимость. Требования задачи могут быть не столь высокими, чтобы применять всю "мощь" встроенного языка.

Во всех языках функции разделяются на группы, часть из которых присутствует практически во всех языках: математические функции, функции работы со строками, обмен по SQL , DDE - обмен и т. д.

В разрабатываемом приложении создаются программные фрагменты, состоящие из операторов и функций языка, которые выполняют некоторую последовательность действий. Эти программные фрагменты связываются с разнообразными событиями в приложении, такими как нажатие кнопки, открытие окна, выполнение логического условия (a +b > c). Каждое из событий ассоциируется с графическим объектом, окном, таймером, открытием/ закрытием приложения. Когда приложение содержит сотни окон, тысячи различных графических объектов, а с каждым из них связано несколько событий, в приложении может "работать" огромное количество отдельных программных фрагментов. Велика вероятность их "одновременной" активизации.

Каждая из функций во встроенном языке выполняется в синхронном или асинхронном режиме. В синхронном режиме выполнение следующей функции не начинается до тех пор, пока не завершилось исполнение предыдущей. При запуске асинхронной функции управление переходит следующей, не дожидаясь завершения исполнения предыдущей функции.

В связи с этим возникает несколько вопросов. С каким приоритетом исполняется каждый из фрагментов, допускается ли рекурсия при обработке событий и если да, то каков уровень вложенности? В SCADA - системах уровень вложенности пока не стандартизован, но оговаривается особо в рамках каждой из них.

5.1. Скрипты в InTouch

Скрипты в InTouch - это программные фрагменты, активизируемые по событиям (по нажатию клавиши, кнопки, открытию окна, изменению значения переменной и т. д.).

5.1.1. Типы скриптов

В InTouch различают несколько типов скриптов:

- Application Scripts (скрипты уровня приложения) относятся ко всему приложению и используются для запуска других приложений, имитации технологических процессов, вычисления значений переменных и т.д.
- Window Scripts (скрипты уровня окна) связываются с конкретным окном.
- Key Scripts (клавишные скрипты) привязываются к какой-либо клавише или комбинации клавиш клавиатуры. Это может быть полезным при создании каких-либо глобальных для всего приложения функций (возврат в главное окно, окончание сеанса работы с приложением и т. д.).
- Touch Pushbutton Action Scripts (скрипты, запускаемые кнопками) очень похожи на клавишные скрипты и связываются с объектами, которые будут использоваться в качестве исполнительных кнопок. Эти скрипты запускаются при каждом нажатии на объект-кнопку.
- Condition Scripts (скрипты по изменению логического выражения) связываются с логической переменной или выражением, которое будет принимать значения либо "истина", либо "ложь". Логические скрипты могут содержать в себе и аналоговые переменные.
- Data Change Scripts (скрипты по изменению данных) связываются либо с переменной, либо с полем переменной. Эти скрипты исполняются только один раз, когда значение

переменной либо поля меняется на величину, превышающую значение допуска, заданного в словаре переменных.

- ActiveX Event (скрипты событий ActiveX) предназначены для поддержки механизма реакции на события в ActiveX - объектах. С каждым событием может быть связан один скрипт типа ActiveX Event, запускающийся в WindowViewer во время исполнения приложения.
- Quick Function - скрипты, которые могут вызываться из других скриптов и использоваться в выражениях при определении динамических свойств объектов.

Диалоги редактора, открываемые при создании скриптов различных типов, имеют небольшие отличия. Вызов диалога редактора скриптов в окне WindowMaker осуществляется командой Special/Scripts с последующим выбором типа создаваемого или редактируемого скрипта. Для этого можно также воспользоваться окном Application Explorer, выбрав папку Scripts. На рис. 5.1.1 приведен диалог Application Scripts (скрипты уровня приложения).

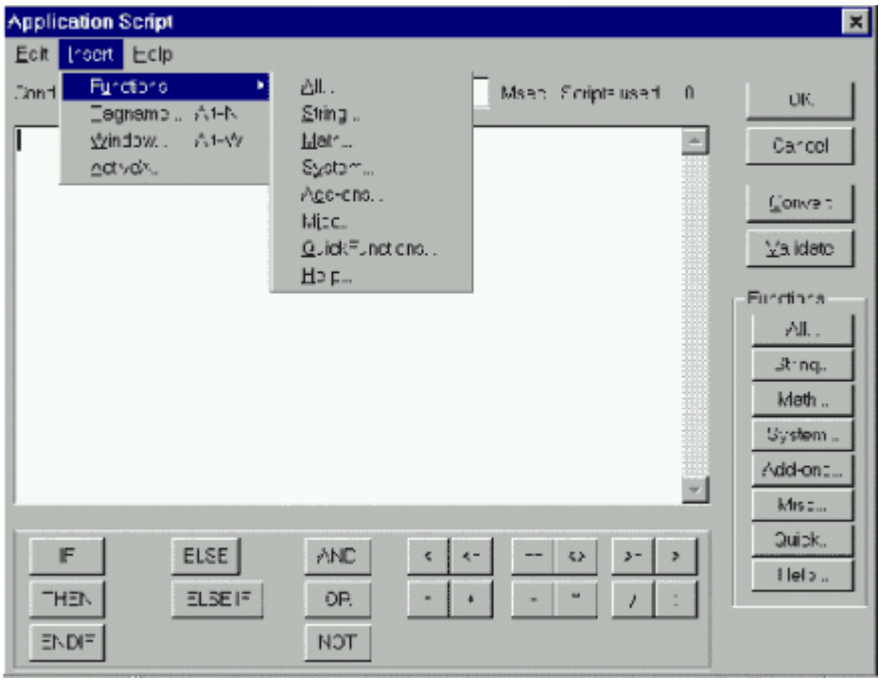


Рис. 5.1.1. Редактор скриптов Application Scripts (уровень приложения).

Редактор скриптов InTouch поддерживает два типа скриптов: простые и сложные. Простые скрипты - это скрипты, содержащие операторы присваивания, сравнения, простые математические функции и т. д. Сложные скрипты позволяют выполнять различные логические операции типа IF - THEN - ELSE, а также могут включать циклы типа FOR - NEXT.

Справа, в поле Functions, размещены клавиши вызова списков различных групп встроенных функций. Доступ к спискам встроенных функций возможен также командой Insert/Functions с последующим выбором группы функций (см. рис. 5.1.1).

5.1.2. Встроенные функции

В пакете InTouch имеется набор встроенных функций, которые могут быть связаны с командами или использованы в скриптах для выполнения самых различных задач.

Все встроенные функции разбиты на четыре группы:

- String... - для обработки различных символьных строк и переменных;
- Math... - математические функции;
- System... - системные функции;

- Misc... - функции для работы с алармами распределенных систем, трендами, печатью и др.

Вызов списка функций группы осуществляется нажатием соответствующей клавиши. Например, щелчок по клавише String... редактора скриптов вызывает появление диалога Choose function (выбор функции) со списком строковых функций (рис.5.1.2).

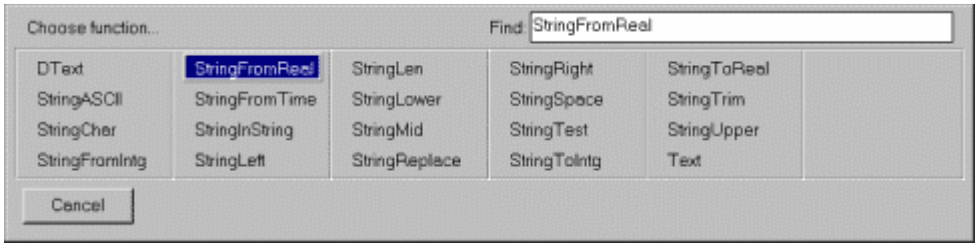


Рис.5.1.2. Список строковых функций.

Описание некоторых функций этого списка приведено в табл. 5.1.

Функция	Описание
StringFromIntg()	Возвращает символьное представление целого аргумента в указанной системе счисления
StringFromReal()	Возвращает символьное представление вещественной величины либо в формате с плавающей запятой, либо в экспоненциальном формате
StringLen()	Возвращает длину указанной строки
StringToIntg()	Преобразует символьное представление целого числа во внутренний формат
StringUpper()	Преобразует все символы исходной строки в нижнем регистре в верхний регистр
Text()	Осуществляет форматированный вывод указанной целой или вещественной переменной в соответствии со строкой форматирования

Таблица 5.1.

Каждая строковая функция имеет один или несколько аргументов (до 6). Например, синтаксис функции StringFromReal выглядит следующим образом:

StringFromReal(Number,Precision,Type);

- Number - конвертируемая вещественная величина;
- Precision - количество десятичных знаков;
- Type - тип формата ("f", "e", "E").

Например,

функция StringFromReal(263.365, 2, "f") возвращает "263.36";
функция StringFromReal(263.365, 2, "e") возвращает "2.63e2";
функция StringFromReal(263.55, 3, "E") возвращает "2.636E2".
Функция Text имеет два аргумента: Text(Analog_Tag, "Format_Text");
- Analog_Tag - вещественное или целое число;
- Format_Text - формат преобразования.

Если указанный формат функции Text - "#0.00", то:

- при Analog_Tag = 66 функция возвращает 66.00;
- при Analog_Tag =22.269 функция возвращает 22.27;
- при Analog_Tag =9.999 функция возвращает 10.00.

- Щелчок по клавише Math... вызывает появление диалога Choose function (выбор функции) со списком математических функций.

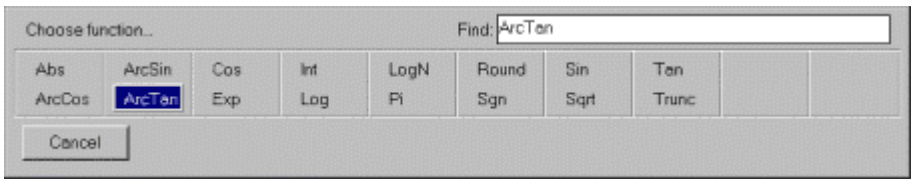


Рис.5.1.3. Список математических функций.

Математические функции работают с целыми и вещественными аргументами, выдавая целый или вещественный результат. В левой части оператора присваивания допускается указывать и целые переменные. Однако необходимо иметь ввиду, что преобразование вещественного значения в целое может привести к усечению результата.

- Системные функции делятся на две категории: файловые (File) и для работы с Windows - приложениями (Info). Список системных функций приведен на рис.5.1.4.

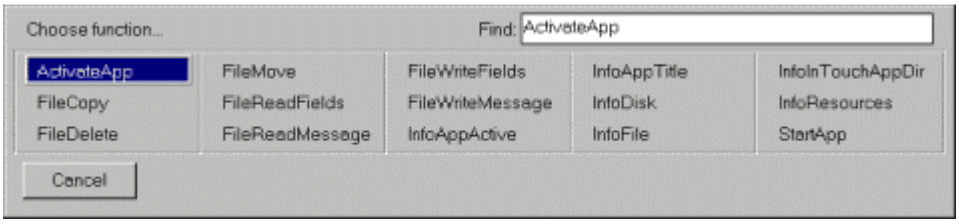


Рис.5.1.4. Список системных функций.

Файловые функции предназначены для считывания и записи информации в файлы. У всех файловых функций есть два общих аргумента - Filename и FillOffset. Аргумент Filename (имя файла) хранит имя файла, из которого должна быть считана или в который должна быть записана информация (имя также должно включать и путь к файлу). Аргумент FillOffset (смещение в файле) задает относительную позицию в файле, начиная с которой будут читаться или записываться данные. Смещение задается в байтах от начала файла. Первый байт файла имеет смещение 0. После завершения каждая функция возвращает следующее доступное смещение в файле. Например, если функция читает 5 байтов данных, начиная с 10-го байта, то после завершения функция возвратит 15. Некоторые встроенные функции группы System приведены в табл. 5.2.

Функция	Описание
FileCopy()	Копирует исходный файл в файл-приемник
FileReadFields()	Возвращает очередную запись данных из CSV - файла
FileReadMessage()	Возвращает указанное количество байтов (или всю строку) из указанного файла
FileWriteFields()	Сохраняет в CSV - файле запись данных, состоящую из разделенных запятыми величин
InfoDisk()	Возвращает информацию об указанном локальном или сетевом диске
InfoFile()	Возвращает информацию об указанном файле или подкаталоге компьютера или сетевого устройства
InfoTouchAppDir()	Возвращает имя текущего каталога InTouch - приложения

Таблица 5.1.

Остальные аргументы файловых функций не поддаются типизации и различны для каждой функции.

Например, функция FileReadFields имеет четыре аргумента и следующий синтаксис:

FileReadFields(Filename,FileOffset,StartTag,NumberOfFields);

- StartTag - идентифицирует первый элемент в имени InTouch-переменной;
- NumberOfFields - идентифицирует число полей для чтения.

- Группа функций Miscellaneous (клавиша Misc...) включает функции для работы с алармами распределенных систем, трендами, печатью и др.

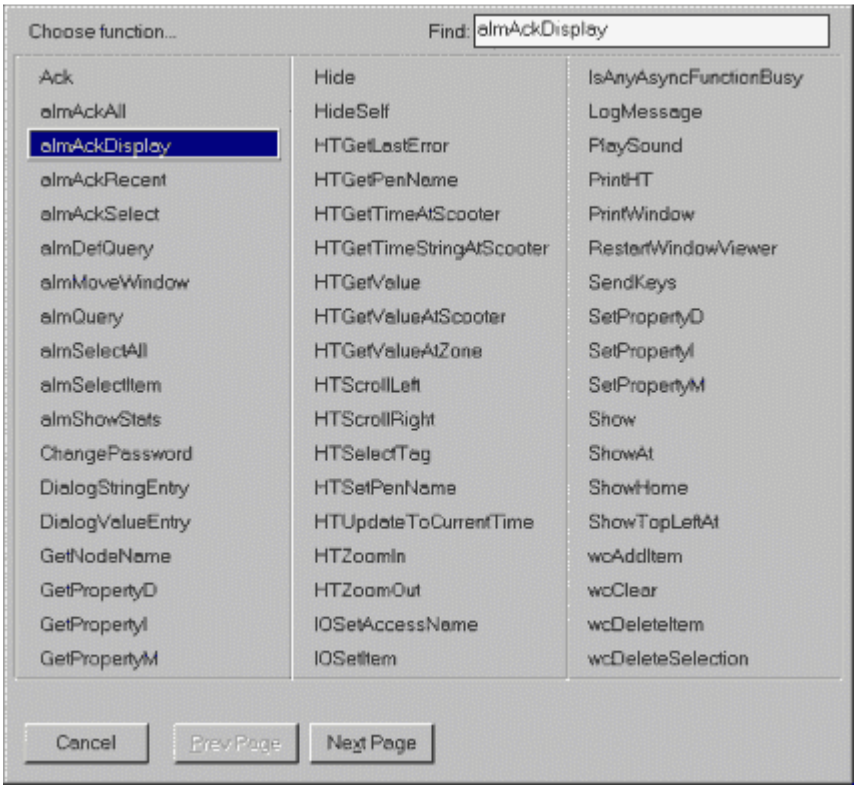


Рис.5.1.5. Список функций группы Miscellaneous.

В этой широкой (с точки зрения назначения функций) группе можно выделить несколько более узко специализированных подгрупп. Функции, название которых начинается с alm, используются только в распределенных системах алармов. Некоторые из них приведены в табл.5.3.1.

Функция	Описание
almAckDisplay()	Подтверждает только те алармы, которые в текущий момент видны в окне отображения алармов
almAckSelect()	Подтверждает алармы, отмеченные оператором в окне отображения алармов
almShowStats()	Выводит панель статистики объекта отображения алармов

Таблица 5.3.1.

Первым аргументом всех встроенных функций алармов является ObjectName (имя объекта алармов). Часто в роли одного из аргументов выступает Comment (комментарий). Например, функция almAckSelect имеет следующий синтаксис: almAckDisplay(ObjectName,Comment);

Функции, название которых начинается с HT, используются только с архивными трендами. Примеры таких встроенных функций - в табл.5.3.2.

Функция	Описание
HTGetPenName()	Возвращает имя переменной, связанной в текущий момент с указанным пером указанного тренда
HTGetValue()	Возвращает значение указанного типа, вычисляемого для указанного пера в пределах всего тренда

HTScrollLeft()	Устанавливает в качестве начала графика более раннее время. Визуально происходит прокрутка тренда влево
HTSetPenName()	Связывает перо тренда с указанной переменной
HTZoomIn()	Масштабирует существующий тренд путем задания новых времени начала и охватываемого интервала времени

Таблица 5.3.2.

Встроенные функции для работы с архивными трендами также могут иметь несколько аргументов (до четырех). Функции, приведенные в табл. 5.3.2, имеют следующий синтаксис:

- HTGetPenName(Hist_Tag, UpdateCount, PenNum);
- HTGetValue(Hist_Tag,UpdateCount,PenNum,ValType_Text);
- HTScrollLeft(Hist_Tag,Percent);
- HTSetPenName(Hist_Tag,PenNum,Tagname);
- HTZoomIn (Hist_Tag,LockString).

Первый аргумент всех встроенных функций для работы с трендами - Hist_Tag (имя тренда). Из других аргументов следует отметить PenNum (номер пера тренда), ValType_Text (строка, указывающая тип возвращаемого значения), Tagname (новое имя пера).

Функции, название которых начинается с wc (табл.5.3.3), используются с управляющими объектами окна (простые списки, текстовые окна, ниспадающие списки и т. д.)

Функция	Описание
wcDeleteItem()	Уничтожает элемент с заданным порядковым номером как в простом, так и в ниспадающем списке
wcInsertItem()	Вставляет указанное сообщение в список
wcLoadText()	Заменяет содержимое текстового окна на новую информацию

Таблица 5.3.3.

Функции этой подгруппы также могут иметь до четырех аргументов:

- wcDeleteItem("ControlName", ItemIndex);
- wcInsertItem("ControlName", ItemIndex, "MessageTag");
- wcLoadText("ControlName", "Filrename");.

Первый аргумент всех встроенных функций этой подгруппы - ControlName (имя управляемого окна). Часто в качестве аргумента используются ItemIndex (номер, соответствующий позиции элемента), MessageTag (строковое сообщение), Filrename (имя файла в формате ASCII).

В рассматриваемой группе функций Miscellaneous следует отметить функцию PrintWindow, i?aaiacia?aiioю для печати окна. Ее синтаксис выглядит следующим образом:

PrintWindow("Window",Left,Top,Width,Height,Options);,

где:

- Window - имя окна;
- Left - число дюймов от левого края;
- Top - число дюймов от верхнего края;
- Width - ширина распечатываемого окна;
- Height - высота распечатываемого окна;
- Options - дискретные значения 0 или 1.

Вставка встроенных функций в скрипт производится щелчком по выбранной функции в списке функций. Она вместе со своими аргументами будет автоматически вставлена в текст

скрипта в точку, указанную курсором. После этого можно отредактировать список аргументов.

По окончании редактирования скрипта следует нажать кнопку Ok. При обнаружении в скрипте каких-либо ошибок на экран будет выведено соответствующее сообщение. В большинстве случаев курсор установится в ту позицию, которая привела к появлению ошибки. Прежде чем скрипт будет сохранен, все ошибки должны быть исправлены.

5.1.3. Функции Quick Function

Quick Functions - это скрипты, которые могут вызываться из других скриптов и использоваться в выражениях при определении динамических свойств объектов. Скрипты Quick Functions хранятся внутри того приложения, в котором они были созданы, и могут многократно использоваться в других скриптах InTouch.

Наиболее часто эти функции используют в выражениях при определении динамических свойств объектов. Чем это вызвано? Дело в том, что длина выражения в поле Expression диалогов определения динамических свойств объектов должна быть не более 256 символов. Это относится к таким динамическим свойствам, как цвет линии, цвет заполнения, изменение высоты и ширины, вертикальное и горизонтальное перемещение, вертикальное и горизонтальное заполнение, видимость, мерцание, ориентация, блокировка.

Диалог Fill Color (цвет заполнения), содержащий поле Expression, приведен на рис.5.1.6.

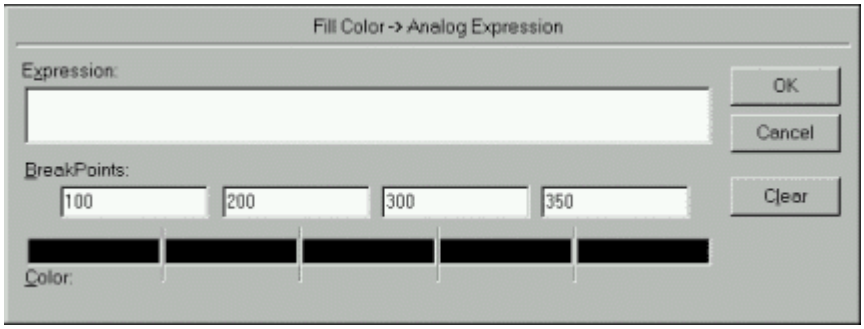


Рис.5.1.6. Диалог Fill Color (цвет заполнения).

Для ввода более длинных выражений можно воспользоваться функциями Quick Functions. При этом выражение в поле Expression должно содержать операторы CALL вызова функций Quick Functions, каждая из которых, в свою очередь, должна иметь в качестве последнего оператора RETURN для возврата результата в вызывающее выражение. Организованное таким образом выражение может содержать многие тысячи символов и быть сколь угодно сложным.

Сохраненная функция Quick Functions может быть использована в любом другом скрипте или выражении.

Quick Functions могут быть синхронными и асинхронными скриптами. Синхронные скрипты выполняются последовательно, в то время, как после запуска одного асинхронного скрипта может быть запущен другой (синхронный или асинхронный) скрипт. Это позволяет отделять исполняющиеся довольно долго операции (типа обращений к базам данных) от основной программы. Асинхронные скрипты не могут возвращать результаты. Поэтому в качестве скриптов Quick Functions, используемых в выражениях (Expression) для определения динамических свойств объектов, следует применять только синхронные скрипты.

Создание скриптов Quick Functions осуществляется в диалоговом окне редактора Quick Functions. Вызов этого диалога на экран в окне WindowMaker производится в командой Special/Scripts с последующим нажатием на строке Quick Functions.

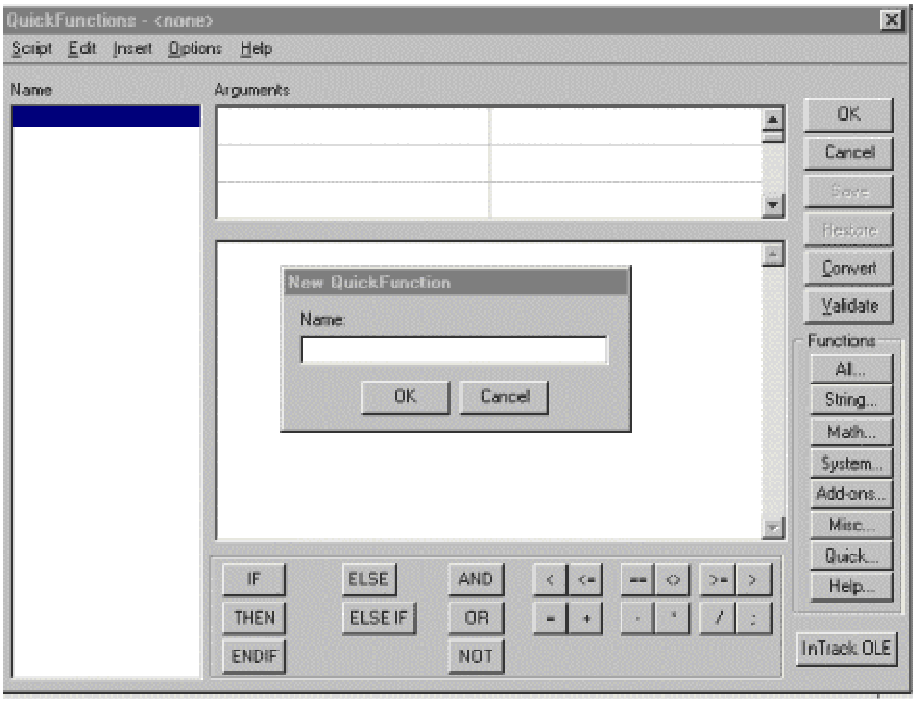


Рис.5.1.7. Диалог редактора Quick Functions.

Список Name содержит имена всех определенных к данному моменту скриптов Quick Functions. Щелчок по имени скрипта выводит его текст в рабочее поле диалога.

Команда Scripts/New предназначена для создания нового скрипта и вызывает на экран диалог для ввода его имени (в середине рис.5.1.7). После щелчка по Ok новое имя будет включено в список имен Name.

Следующий этап - определение аргументов нового скрипта в таблице Arguments диалога Quick Function. В левую колонку таблицы вводят имя аргумента (до 31 символа), в правую - его тип (Integer, Real, Discrete, Message). В одном скрипте допускается до 16 аргументов.

После определения типов аргументов можно приступать к написанию текста скрипта Quick Function в рабочем поле (под таблицей Arguments).

5.2. Встроенный язык программирования Cicode

Cicode - встроенный язык программирования системы Citect, созданный специально для мониторинга и управления приложениями. Это структурированный язык, похожий на Visual Basic или 'C'. Применение Cicode предоставляет пользователю доступ к данным проекта в режиме реального времени, а также ко всем переменным, алармам, трендам, отчетам и т. д.

Cicode поддерживает многозадачность и удаленный вызов процедур.

5.2.1. Команды Cicode

Для управления системой Citect и технологическим процессом используются команды. Каждая команда имеет механизм ее запуска. Команды могут быть вызваны вручную, когда оператор нажмет некоторую последовательность клавиш или кнопку на графической странице. Можно произвести конфигурирование команд для автоматического выполнения:

- при регистрации оператора для входа или выхода из среды исполнения;
- при открытии и закрытии графических страниц;
- при срабатывании алармов;
- при срабатывании событий;
- при выдаче отчетов.

Наиболее часто используют два типа команд:

- Touch commands (команды по нажатию) - активируются путем щелчка мышью на объекте.
- Keyboard commands (команды клавиатуры) - активируются путем набора соответствующих инструкций с клавиатуры.

Команды по нажатию (Touch Commands).

Оператор может выполнять команду (или серии команд) щелчком мыши на объекте. Можно задать несколько команд для одного объекта: одна команда выполняется, когда оператор нажал клавишу мыши на объекте, другая - когда отпустил клавишу, и третья - если оператор нажал и удерживает клавишу мыши. Предоставляется также возможность определить запрещающее условие для любого объекта на странице (включая кнопки). Когда это условие активно, объект не выделен или даже скрыт, и оператор не может его выбрать.

Можно привести множество примеров применения Touch Commands. В графических интерфейсах часто используют кнопки для запуска и остановки насосов, для вкл./выкл. электродвигателей, для перехода на другие графические страницы. Характерным примером применения Touch Commands является вызов выпадающего окна для ввода информации (суперджин). В одних случаях это лицевая панель контроллера, в других - пульт управления насосом или клапаном. С помощью кнопок и иконок, расположенных в этих выпадающих окнах, можно выполнять различные команды: щелчок по соответствующей иконке вызывает смену режима работы контроллера (ручное и автоматическое управление), клавишами "пуск" и "стоп" оператор включает и выключает насосы и т. д.

Команды клавиатуры.

Команды клавиатуры - это команды или серии команд, активируемые при введении оператором определенной последовательности клавиш.

Можно описывать команды клавиатуры, которые будут действовать:

- на всех графических страницах (System Keyboard commands - задаются в Project Editor);
- только на определенной графической странице (Page Keyboard commands - задаются в Page properties);
- только, если оператор указал мышью на определенный объект (Object keyboard commands - задаются в свойствах объекта).

Если одна и та же последовательность клавиш назначена для различных команд в зависимости от местонахождения, будет исполняться команда с максимальным приоритетом. Порядок приоритета (от высшего к низшему) следующий:

- объектные команды клавиатуры;
- страничные команды клавиатуры;
- общесистемные команды клавиатуры.

Командам можно присвоить привилегии и посылать сообщения на регистрацию команды и времени ее подачи.

Для определения команд необходимо ввести выражение или несколько выражений в поле команд закладки Input диалога Свойства объекта (рис.5.2.1). Каждое выражение в команде обычно используется для решения одной задачи, такой как ввод значения переменной, вычисление значения, вывод сообщения на экран, запуск отчета и т. д.

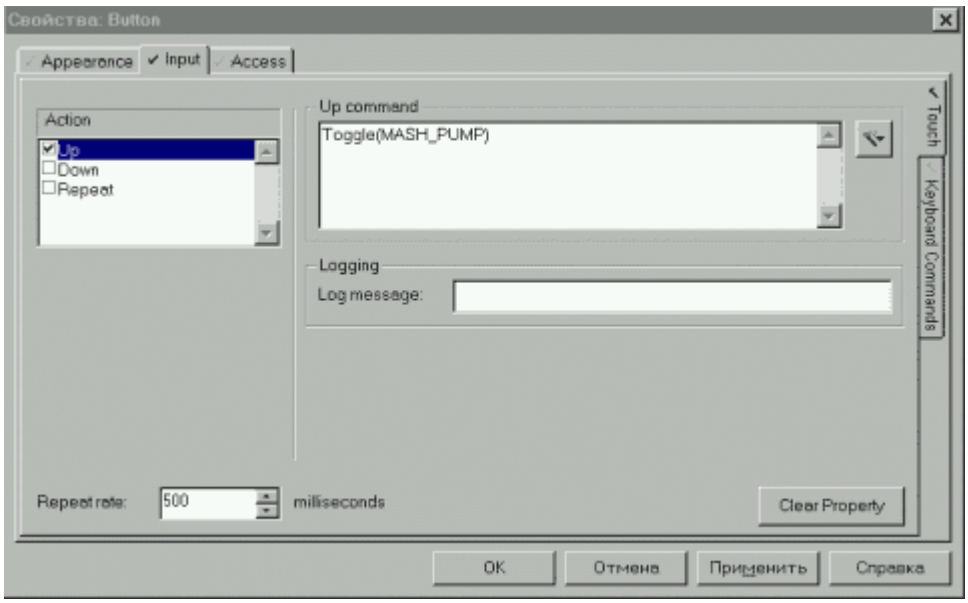


Рис.5.2.1. Диалог Свойства объекта на закладки Input/Touch.

5.2.2. Выражения Cicode

Cicode - выражения являются базовыми элементами языка Cicode. В выражениях могут быть константы, значения переменных или результаты сложных вычислений. Выражения можно использовать для вывода на экран или регистрации данных для мониторинга и анализа, для запуска различных состояний системы, таких как алармы, события, отчеты.

В отличие от команд, выражения не выполняют конкретных задач, они их оценивают. Этот процесс оценки значения можно использовать для вывода информации на экран или принятия решений.

5.2.3. Функции Cicode

Cicode - функции могут выполнять более сложные задачи, чем команды и выражения. Citect имеет около 700 встроенных функций, которые могут показывать страницы, подтверждать алармы, делать вычисления и т. д.

- Cicode - функция - это набор выражений, переменных, операторов, условий выполнения и других функций. Эти функции эквивалентны подпрограммам BASIC и подпрограммам или функциям, используемым в Pascal или C.
- Вызов функции осуществляется введением ее имени в любую команду или выражение. При этом должен быть соблюден следующий синтаксис:
  FunctionName(Arg1, Arg2,...); где:
  - FunctionName - имя функции;
  - Arg1, Arg2,... - аргументы функции.
- Обычно функции требуют нескольких аргументов, но некоторые функции имеют один строковый аргумент. Например, функция PageDisplay("Boiler 1"); вызывает графическую страницу "Boiler 1". Следует обратить внимание на то, что строковый аргумент помещен в двойные кавычки.

Большинство функций требует нескольких аргументов. Список аргументов должен находиться в скобках, один от другого аргументы отделяются запятой. Очень важен порядок введения аргументов в функцию.

Например, функция Login("Manager", "ABC"); предназначена для регистрации пользователя в системе. Первый аргумент ("Manager") указывает имя пользователя, а второй аргумент ("ABC") - его пароль. Если изменить порядок ввода аргументов, при регистрации

пользователя будет выведена ошибка. В качестве аргумента можно использовать целые и действительные числа.

В качестве примера такой функции можно привести AlarmAck(1, 0);. С помощью этой функции можно подтверждать алармы на страницах текущих алармов. Первый аргумент несет информацию о выбранном способе подтверждения алармов (1 - подтверждение всех алармов страницы), а второй - о списке алармов на странице (0 - подтверждение всех алармов списка, на котором установлен курсор).

Возможно использование в качестве аргументов встроенных функций и переменных проекта. Например, функция DspStr(25, "TextFont", B1_TIC_101_PV); выводит значение переменной B1_TIC_101_PV в анимационной точке AN25. Если заключить переменную в двойные кавычки ("B1_TIC_101_PV"), то будут выведены текстовые символы B1_TIC_101_PV, а не значение переменной. Второй аргумент несет информацию о шрифте, которым будет осуществлен вывод.

В качестве аргумента в функции можно использовать последовательность клавиш, вводимую оператором в режиме исполнения. Например, для облегчения выбора оператором страниц проекта можно определить последовательность клавиш ####### Enter и команду PageDisplay(Arg1);.

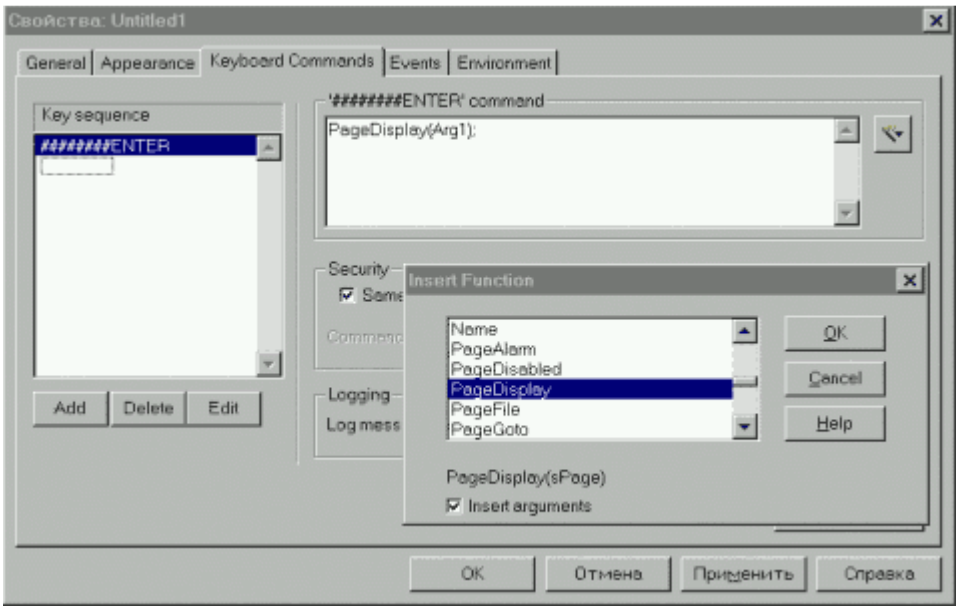


Рис.5.2.2. Диалог Свойства страницы на закладке Keyboard Commands.

При выполнении команды вводом с клавиатуры имени страницы с последующим нажатием клавиши Enter ее имя вводится в качестве аргумента Arg1 функции PageDisplay. Таким образом, оператор может вывести на экран любую страницу проекта.

- Имя каждой функции (FunctionName) включает следующую информацию:
  - от трех до пяти букв для типа функции - Trend, Plot, Win...;
  - одно или два слова описания данных - Info, ClietInfo, Mode...;
  - одно слово описания действия - Get, Set, Read...
- Синтаксис встроенных функций может быть представлен следующим образом:

<Scope>
<ReturnDataType>
FUNCTION <functionname>
(<arg1datatype arg1[=DefaultValue]>,
<arg2datatype arg2[=DefaultValue]>,
. . .
<argndatatype argn[=DefaultValue]>)

```
<Statement(s);>
:
RETURN <ReturnValue;>
END
```

Во встроенной функции можно выделить семь основных частей :

- Scope - область применения (Public - для всех файлов или только объявленных в функции - Private);
- DataType - тип данных всех аргументов отдельной строкой;
- слово FUNCTION, набранное на клавиатуре отдельной строкой;
- FunctionName - имя функции; - аргумент (список аргументов, отделенных запятой);
- Statement(s) - выражение/выражения отдельной строкой в программе;
- слово END.

По умолчанию, область применения функции - Public. Другими словами, эта функция будет доступна всем файлам Cicode, страницам и базам данных проекта. Если функция объявлена как Private, то она доступна только тому файлу, в котором объявлена.

Обязательным является объявление типа всех аргументов функции (INT, REAL, STRING, OBJECT).

В системе Citect насчитывается несколько десятков групп встроенных функций. Основные из них приведены в табл. 5.4.

Группа функций	Описание
ActiveX	Вызывает и взаимодействует с ActiveX - объектами
Alarm	Управляет алармами
Communication	Обеспечивают доступ к коммуникационным портам
DDE	Обеспечивают обмен данными между Citect и другими Windows - приложениями
Display	Управляют графическими страницами
DLL	Осуществляют функции в библиотеке динамических связей
File	Обеспечивают доступ к стандартным ASCII - файлам
Group	Манипулируют группами зон, устройств, категорий алармов
I/O Device	Управляют устройствами В/В
Math/Trig	Стандартные математические и тригонометрические функции
Miscellaneous	Смешанные функции
Page	Управляют выводом графических страниц, страниц стандартных алармов и трендов
Report	Запускают выдачу отчетов с серверов отчетов
Security	Управляют входом, выходом и правами доступа
SPC	Извлекают SPC - информацию и управляют свойствами и параметрами SPC - вычислений
SQL	Определяют, манипулируют и управляют данными в БД SQL и других реляционных БД
String	Строковые функции
Time/Date	Манипулируют временем и датами переменных
Trend	Управляют трендами
Window	Управляют окнами

Таблица 5.4.

Наиболее часто в Citect используются следующие шесть групп функций: Alarm, Page, Keyboard, Report, Time/date, Miscellaneous (функции для работы с алармами, страницами проекта, клавиатурой, отчетами, временем/датой и смешанные функции).

Группа Alarm включает более 40 встроенных функций. Некоторые достаточно часто применяемые функции этой группы представлены в табл.5.5.

Функция	Описание
AlarmAck(Mode, Value)	Подтверждает аларм
AlarmComment(sComment)	Добавляет комментарий в страницу сводки алармов в режиме исполнения
AlarmDisable(Mode, Value)	Блокирует аларм
AlarmEnable(Mode, Value)	Возвращает доступ к аларму
AlarmHelp()	Вызывает на экран справочную страницу

Таблица 5.5.

Группа Page насчитывает более 20 функций, среди которых часто применяются следующие функции (табл.5.6).

Функция и аргументы	Описание
PageAlarm(Category)	Выводит страницу текущих алармов
PageDisabled(Category)	Выводит блокированные алармы
PageDisplay(Page)	Выводит новую страницу на экран
PageFile(sName)	Выводит файл на странице файлов
PageHardware()	Выводит страницу аппаратных алармов
PageLast()	Выводит страницу, которая предшествовала выведенной в настоящий момент
PageNext()	Выводит предыдущую страницу в соответствии с порядком размещения страниц в проекте.
PagePrev()	Выводит следующую страницу в соответствии с порядком размещения страниц в проекте.
PageSummary(Category)	Выводит страницу сводки алармов
PageTrend(sPage, sTag1 ... sTag8)	Выводит страницу трендов

Таблица 5.5.

Аргументы приведенных в табл. 5.6 функций имеют следующий смысл:

- Category - номер категории аларма;
- Page - имя страницы или ее номер (в двойных кавычках);
- sName - имя файла.

5.2.4. Редактор Cicode

Citect поддерживает около 700 встроенных функций. Эти функции (или комбинация нескольких функций) могут обычно выполнять большинство задач системы управления. Если же некоторые задачи не могут быть решены с помощью встроенных функций, можно написать собственные функции.

Редактор Cicode специально предназначен для редактирования и отладки Cicode - функций. Вход в Редактор Cicode (Cicode Editor) осуществляется из окна Project Editor нажатием иконки в инструментальной панели или командой Tools/ Cicode Editor.

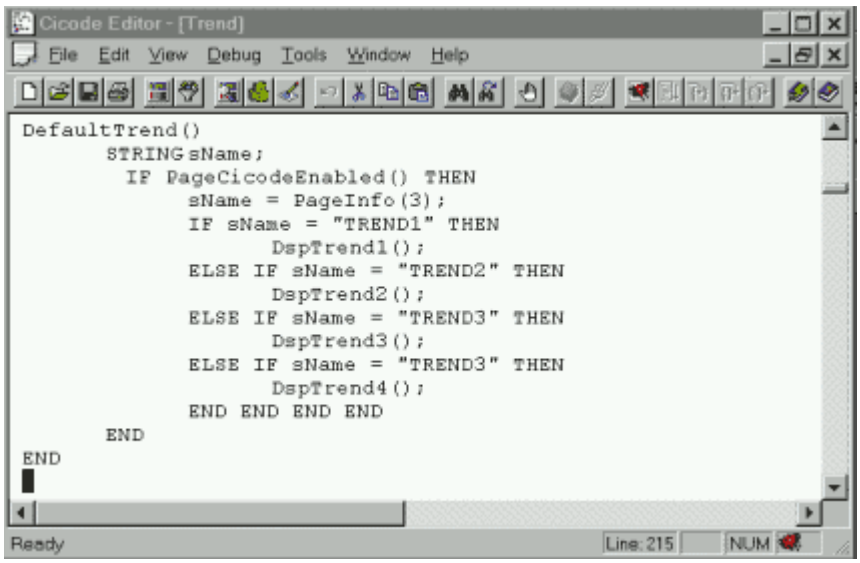


Рис.5.2.3. Окно Редактора Cicode (Cicode Editor).

Cicode Editor - полнофункциональная интегрированная среда программирования для создания и отладки программ на языке Cicode. Рабочая область редактора - окна, куда выводится содержимое файлов с программами на языке Cicode (рис.5.2.3). Одновременно может быть открыто несколько таких окон с программами, принадлежащими различным проектам.

Опции меню содержат все команды, необходимые для создания, редактирования и отладки программ. Для обеспечения быстрого доступа к некоторым командам под строкой меню размещены инструментальные панели, содержащие иконки для редактирования, компилирования и отладки программ.

В Cicode представлены стандартные операторы, применяемые в большинстве языков программирования: математические, операторы отношения, логические, битовые.

Условный оператор IF может быть использован в двух форматах: IF THEN и IF THEN ELSE. Cicode поддерживает два типа операторов цикла: FOR... DO и WHILE... DO.

Встроенные в редактор средства отладки программ позволяют запускать и останавливать процесс отладки, вставлять и удалять точки останова, а также контролировать пошаговое исполнение. Для задания точек останова можно использовать специальную функцию Cicode, либо вставлять их вручную.

Отличительный знак редактора Cicode - " жучок " в правом нижнем углу экрана, меняющий свой цвет в зависимости от режима работы: в режиме редактирования программы - красного цвета, в режиме отладки - зеленого.

Cicode - функции записываются в файлы, хранящиеся на жестком диске. Эти файлы имеют расширение .CI для их идентификации. При компилировании проекта компилятор читает все файлы, в которых хранятся Cicode - функции. Для каждой функции или группы функций можно создать свой файл. С точки зрения обслуживания в одном и том же файле лучше хранить функции, относящиеся к одной задаче. Например, хранить все функции, относящиеся к алармам, в файле Alarm.CI. Файлы с Cicode - функциями хранятся в одной директории вместе с проектом.

5.3. Взгляд со стороны на языки программирования InTouch и Citect

Выносимые на суд читателя системы - InTouch и Citect - предлагают пользователю языки программирования двух типов (см. начало главы 5).

- В основную поставку InTouch входит набор до 100 функций. Но следует отметить, что:

- существуют десятки дополнительных библиотек с InTouch - функциями, которые загружаются отдельно;
- в InTouch возможна разработка Quick - функций на базе имеющихся операторов, встроенных функций и ранее созданных Quick - функций (после сохранения Quick - функции она автоматически появляется в общем списке функций InTouch);
- возможна разработка новых функций с использованием FactorySuite Toolkit и Visual C/C++.

- Язык Cicode в Citect разработан на базе C/C++. Набор встроенных функций в системе превышает 700. Разработка новых функций производится способом, свойственным традиционным языкам программирования.
- Синтаксический анализ программного кода в редакторе скриптов системы InTouch осуществляется в момент сохранения скрипта. При наличии ошибок диалог редактора скриптов не закрывается кнопкой Ok до тех пор, пока все ошибки не будут исправлены. При этом курсор каждый раз указывает на первую ошибку в списке.
- В Cicode синтаксический анализ программы выполняется на этапе компиляции файла Cicode. В этом языке используются свойственные традиционным языкам средства отладки: точки останова, пошаговое исполнение и т. д.
- В InTouch существуют функции для отладки, которые позволяют выводить в специальный файл (Wonderware Logger) статусную информацию о выполнении скриптов.
- Использование Cicode требует более квалифицированной подготовки разработчиков приложений, особенно, если планируется создание многочисленных дополнительных функций.

ГЛАВА 6. БАЗЫ ДАННЫХ

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).

Критерии оценки БД

Базы данных будут продолжать развиваться, а объемы информации в компьютерах - расти. Усложнение производственных процессов, "интеллектуа-лизация" 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?

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.

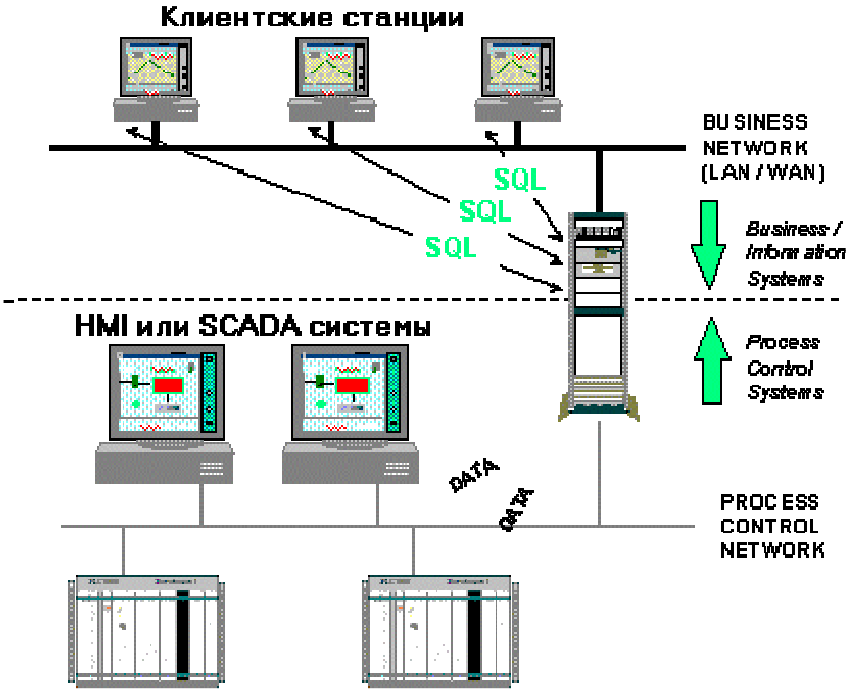


Рис. 6.1. Клиент-серверная организация.

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.

6.1. 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 FactorySuite200. 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.

6.1.1. 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 некритичную 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.

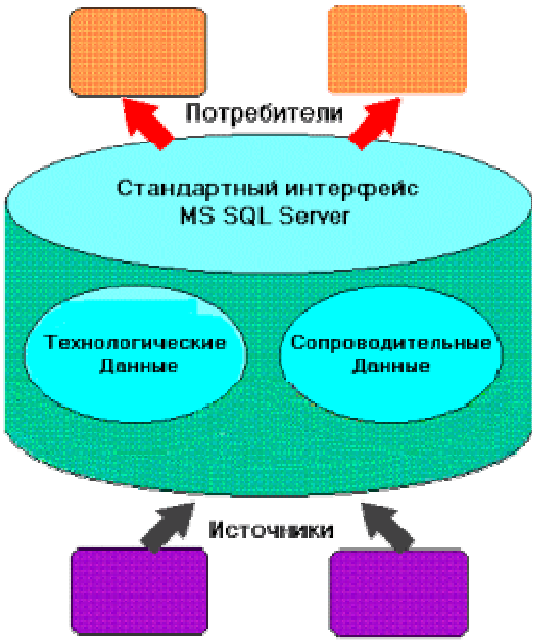


Рис. 6.1.1. IndustrialSQL Server на основе MS SQL Server.

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.

6.1.2. 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/выводы 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.

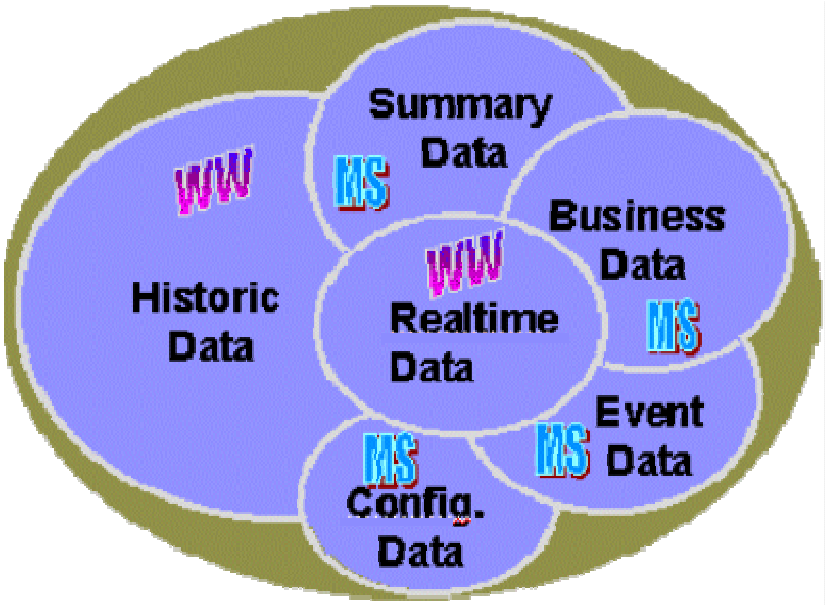


Рис. 6.1.2 Типы данных, регистрируемых IndustrialSQL Server

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.

6.1.3. 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 неустановившемся 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 анормальное поведение процесса. IndustrialSQL Server хранит как оперативные, так и архивные данные и позволяет сравнивать их.

6.2. 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.

6.2.1. 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)

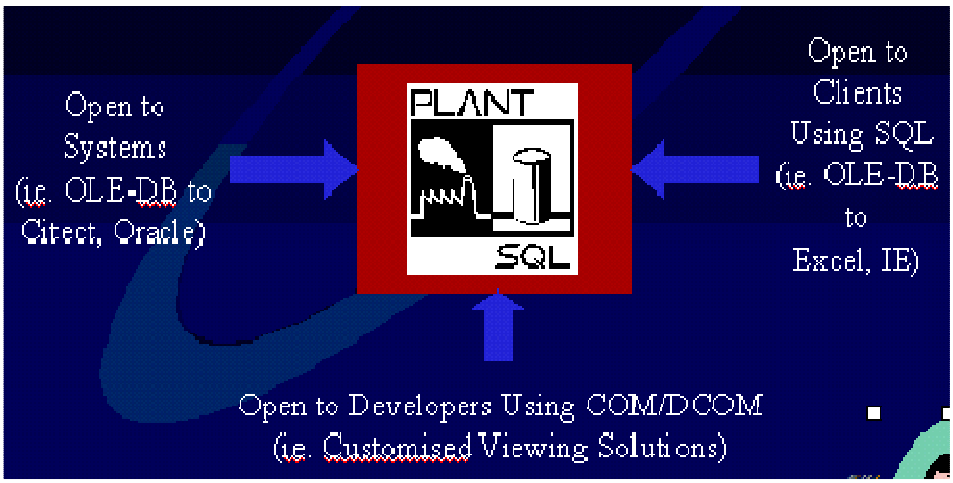


Рис. 6.2.1 Протоколы доступа к Plant2SQL

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).

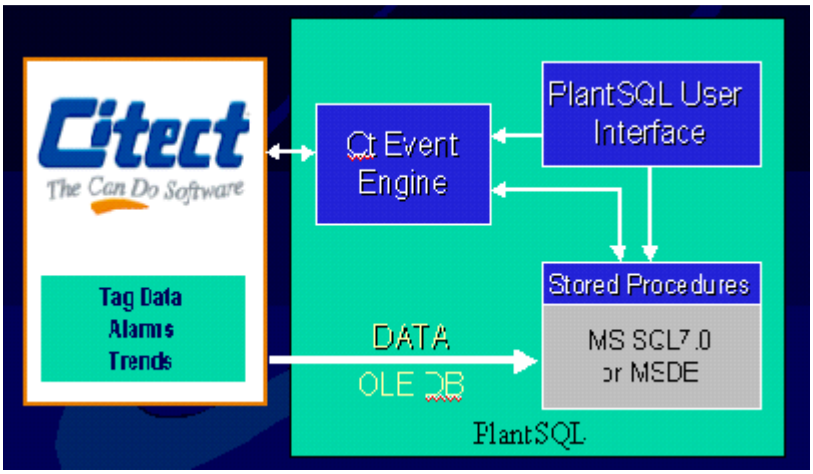


Рис.6.2.2 MS SQL Server - основа Plant2SQL

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.

6.2.2. Область применения

Интеграция заводских данных с бизнес-информацией открывает большие возможности для улучшения деятельности предприятия, качества и производительности.

Персонал отдела качества (Quility 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.

6.3 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.

ГЛАВА 7. INTERNET/INTRANET-РЕШЕНИЯ И SCADA-СИСТЕМЫ.

СТРАТЕГИЯ КЛИЕНТСКИХ ПРИЛОЖЕНИЙ

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.

7.1 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.

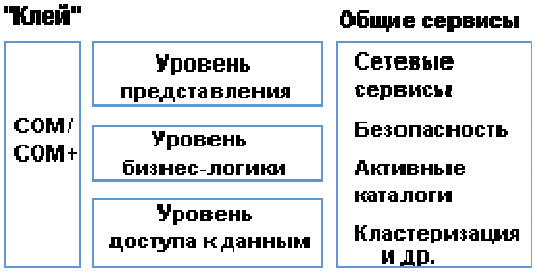


Рис. 7.2. Структра Windows DNA.

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.

Слои 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-броузер.

Далее появилась идея обогащения Web-приложений различными компонентами, которые могут использоваться броузером, - 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).

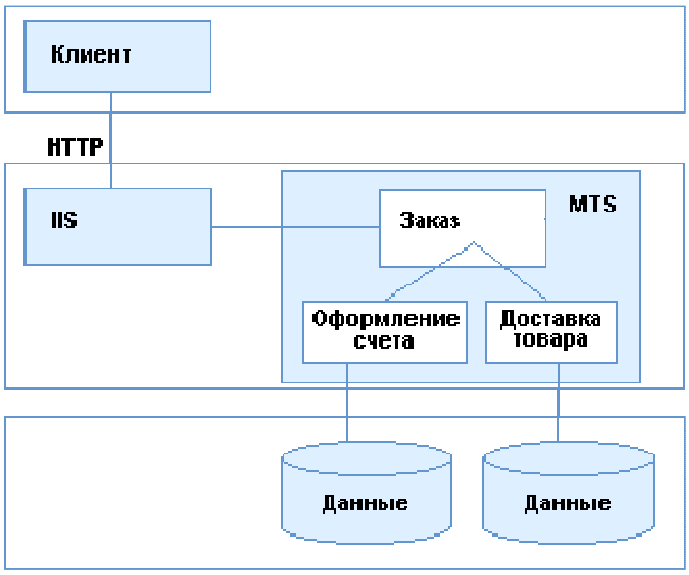


Рис. 7.3. Трехуровневое приложение.

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.

7.2 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 Achitecture), 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).

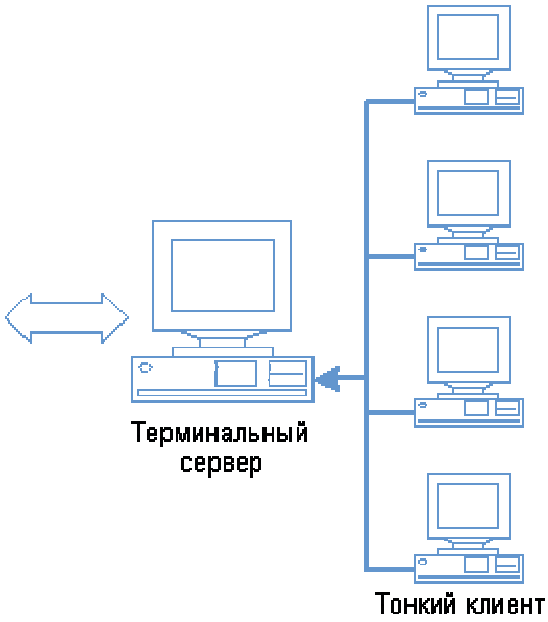


Рис. 7.4. Архитектура терминал-сервер.

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.

7.3 Стратегия клиентских приложений от Wonderware

7.3.1 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.

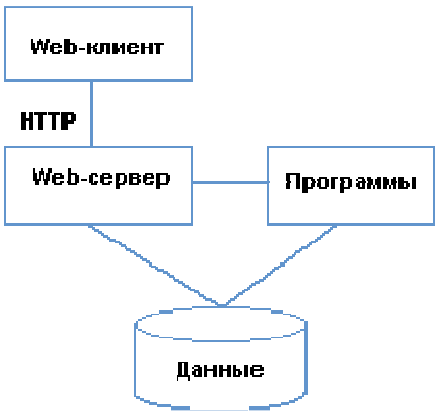


Рис. 7.5. Клиенты и серверы Web.

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-сервврвом 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).

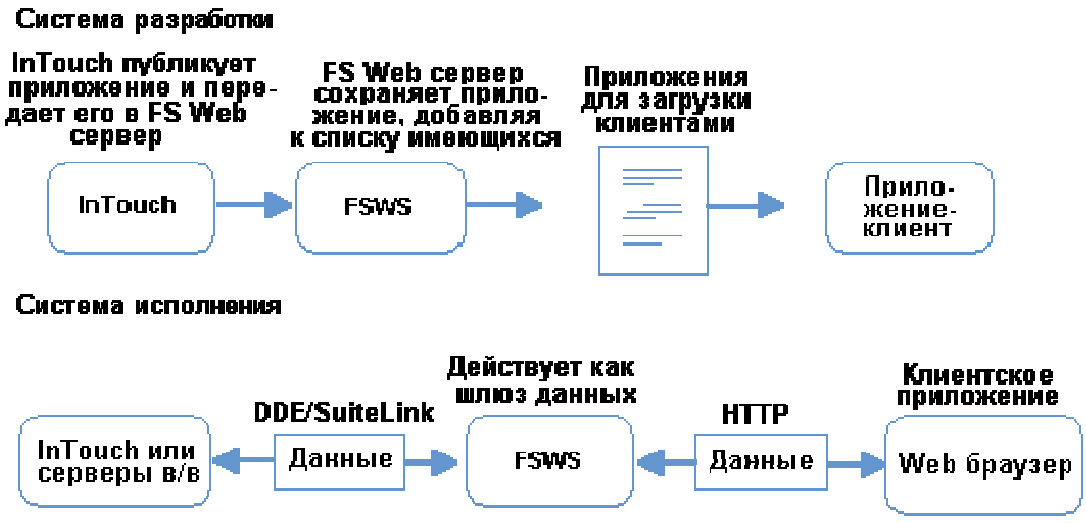


Рис. 7.6. Web сервер для обмена данными между приложениями InTouch.

На рисунке 6 показаны возможности разработки Internet-приложений и запуск их в реальном времен 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.

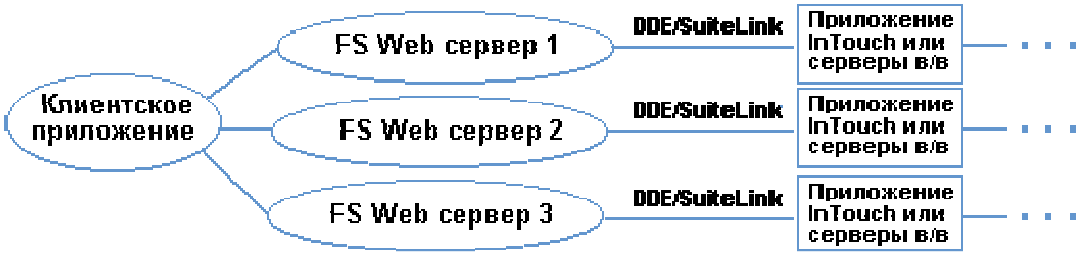


Рис. 7.7. Получение данных от нескольких Web-серверов.

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).

7.3.2 Базы данных реального времени (БДРВ) и Internet-решения

Поскольку БДРВ поддерживают язык SQL-запросов, то для организации доступа с 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.

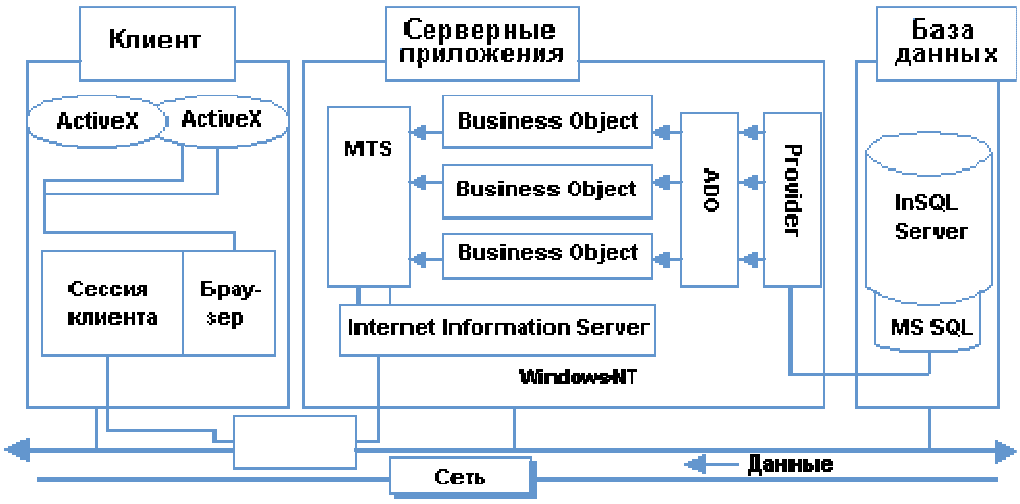


Рис. 7.8. Трехуровневая клиент-серверная архитектура.

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.

7.3.3 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 - броузера, 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 платформно-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 навигатор 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 и 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 навигаторов (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 поставляет 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).

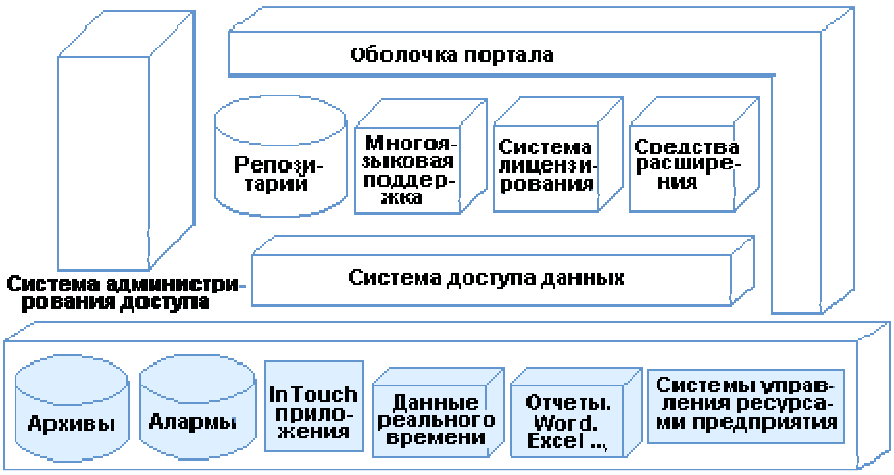


Рис. 7.9. Структура портала SuiteVoyager.

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).

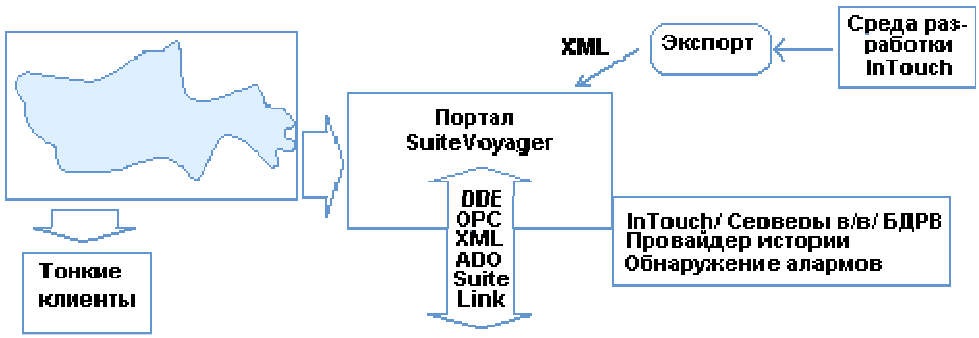


Рис. 7.10. Решение на основе SuiteVoyager.

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" технологии, такие как XML (eXtensible Mark-up-Language).

7.4 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.

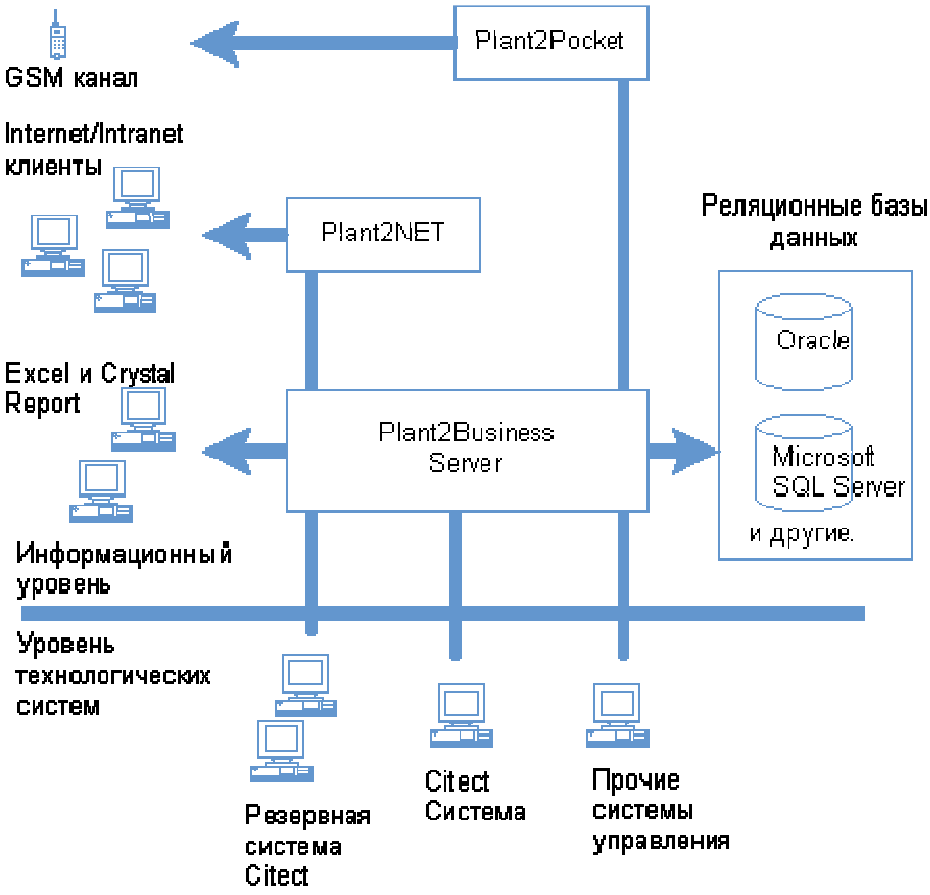


Рис. 7.11. Схема информационных потоков

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.

7.5. 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-протколу для глобальной сети