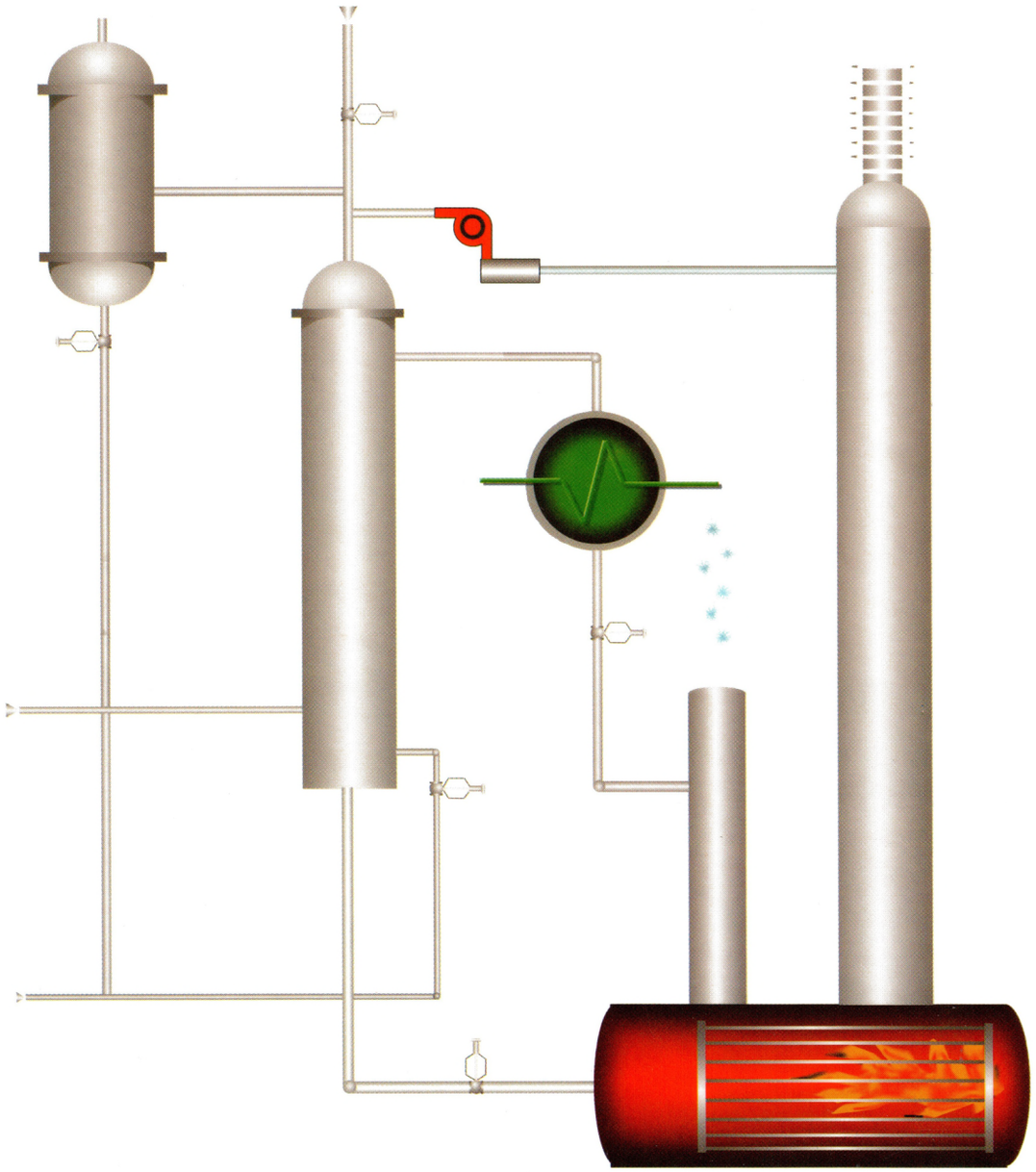
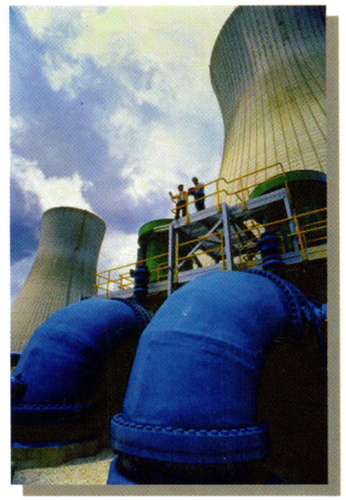


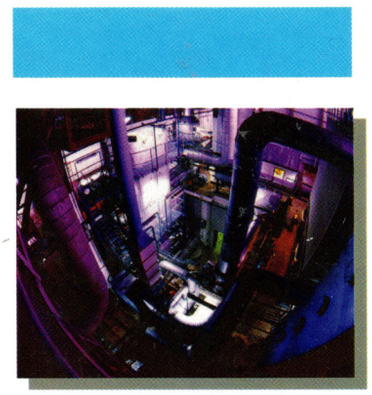
ISBN 5-9900271-1-7

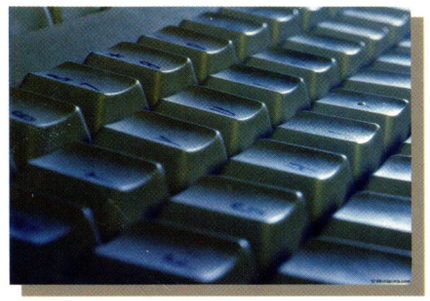
SCADA-системы: взгляд изнутри

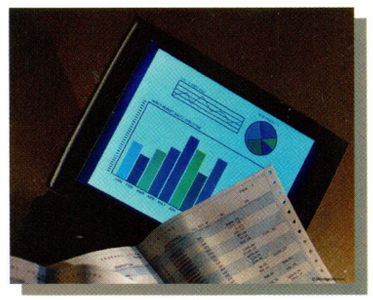
Е. Б. Андреев, Н. А. Куцевич, О. В. Синенко

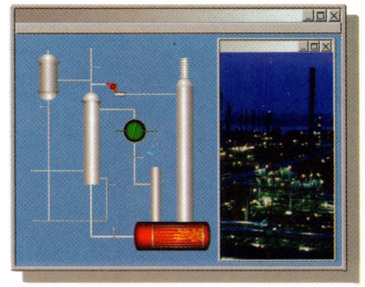
SCADA-системы: взгляд изнутри











Андреев Е.Б., Куцевич Н.А., Синенко О.В.

SCADA-системы: взгляд изнутри

Москва 2004

УДК 004.9SCADA:681.5(075.4)
ББК 32.973.26-018.2
А65

Андреев Е.Б., Куцевич Н.А., Синенко О.В.

А65 SCADA-системы: взгляд изнутри / Е.Б. Андреев, Н.А. Куцевич, О.В. Синенко – М.: Издательство «РТСофт», 2004. – 176 с.: ил.

ISBN 5-9900271-1-7

В книге на примере двух известных и хорошо зарекомендовавших себя SCADA-систем – InTouch и Citect – подробно рассмотрены основные компоненты, функции и возможности систем диспетчерского управления и сбора данных. Книга предназначена для руководителей и специалистов предприятий, которые являются постоянными пользователями SCADA-приложений, и для тех, кто только начинает работать в области промышленной автоматизации. Книга будет также полезна и в учебном процессе при подготовке квалифицированных инженерных кадров, поскольку в настоящее время в учебные планы ряда технических университетов вводятся дисциплины, связанные с изучением SCADA-систем.

УДК 004.SCADA:681.5(0.75.4)
ББК 32.973.26-0.18.2

ISBN 5-9900271-1-7

Содержание

От авторов

Современная автоматизация – это и персональные компьютеры, и контроллеры, и промышленные сети, и, конечно же, программное обеспечение. Программное обеспечение SCADA занимает в этом ряду особое место. Не будет преувеличением сказать, что в последнее десятилетие к этому программному продукту было приковано, пожалуй, самое пристальное внимание специалистов в области автоматизации. И тем более странно, что мало кто из них рискнул высказать свое мнение по этому вопросу (за исключением нескольких десятков статей в специализированных журналах). А ведь обобщенная информация о SCADA-системах очень нужна и специалистам на местах, и новому поколению, делающему первые шаги в мире автоматизации.

Книга «SCADA-системы: взгляд изнутри» – это попытка авторов систематизировать разрозненную информацию об этом программном продукте.

Во введении можно найти ответы на вопросы, связанные с реализуемыми функциями, характеристиками или критериями выбора SCADA-систем.

В последующих пяти главах на примерах двух популярных в России систем – InTouch и Citect – подробно рассматриваются такие базовые функции SCADA, как графический интерфейс, организация взаимодействия с контроллерами, алармы и события, тренды, встроенные языки программирования. Шестая глава посвящена базам данных реального времени. Решения по организации доступа к информации с помощью клиентских приложений, в том числе и базирующихся на Internet-технологии, описаны в седьмой главе.

Заключение – это взгляд на настоящее и будущее автоматизации промышленных предприятий – интеграцию всех уровней управления.

Авторы надеются увидеть среди читателей книги и опытных специалистов-разработчиков, пожелавших сверить свой профессиональный взгляд на SCADA-системы с авторским, и молодежь, чья жизнь в автоматизации уже невозможна без SCADA. Книга также будет полезна руководителям и специалистам предприятий, которые являются постоянными пользователями SCADA-приложений на своих автоматизированных рабочих местах. Многочисленной армии операторов/диспетчеров книга поможет взглянуть на интерфейс с системой управления изнутри, глазами разработчика, а значит, и преодолеть психологический барьер.

Введение

Современная АСУТП (автоматизированная система управления технологическим процессом) представляет собой многоуровневую человеко-машинную систему управления. Создание АСУ сложными технологическими процессами осуществляется с использованием автоматических информационных систем сбора данных и вычислительных комплексов, которые постоянно совершенствуются, по мере эволюции технических средств и программного обеспечения.

АСУТП и диспетчерское управление

Непрерывную во времени картину развития АСУТП можно разделить на три этапа, обусловленные появлением качественно новых научных идей и технических средств. В ходе истории меняется характер объектов и методов управления, средств автоматизации и других компонентов, составляющих содержание современной системы управления.

- **Первый этап** отражает внедрение систем автоматического регулирования (САР). Объектами управления на этом этапе являются отдельные параметры, установки, агрегаты. Решение задач стабилизации, программного управления, слежения переходит от человека к САР. У человека появляются функции расчета задания и параметров настройки регуляторов.

- **Второй этап** – автоматизация технологических процессов. Объектом управления становится рассредоточенная в пространстве система. С помощью систем автоматического управления (САУ) реализуются все более сложные законы управления, решаются задачи оптимального и адаптивного управления, проводится идентификация объекта и состояния системы. Характерной особенностью этого этапа является внедрение систем телемеханики в управление технологическими процессами. Человек все больше отдаляется от объекта управления, между объектом и диспетчером выстраивается целый ряд измерительных систем, исполнительных механизмов, средств телемеханики, мнемосхем и других средств отображения информации (СОИ).

- **Третий этап** – автоматизация систем управления технологическими процессами – характеризуется внедрением в управление технологическими процессами вычислительной техники. Вначале – применение микропроцессоров, использование на отдельных фазах управления вычислительных систем; затем – активное развитие человеко-машинных систем управления, инженерной психологии, методов и моделей исследования операций и, наконец, – диспетчерское управление на основе автоматических информационных систем сбора данных и современных вычислительных комплексов.

От этапа к этапу меняются и функции человека (оператора/диспетчера), призванного обеспечить регламентное функционирование технологического процесса. Расширяется круг задач, решаемых на уровне управления. Ограниченный прямой необходимостью управления технологическим процессом набор задач пополняется качественно новыми задачами, ранее имеющими вспомогательный характер или относящимися к другому уровню управления.

Диспетчер в многоуровневой автоматизированной системе управления технологическими процессами получает информацию с монитора ЭВМ или с электронной системы отображения информации и воздействует на объекты, находящиеся от него на значительном расстоянии, с помощью телекоммуникационных систем, контроллеров, интеллектуальных исполнительных механизмов.

Основой, необходимым условием эффективной реализации диспетчерского управления, имеющего ярко выраженный динамический характер, становится работа с информацией, т.е. процесс сбора, передачи, обработки, отображения, представления информации.

От диспетчера уже требуется не только профессиональное знание технологического процесса, основ управления, но и опыт работы в информационных системах, умение принимать решение (в диалоге с ЭВМ) в нештатных и аварийных ситуациях и многое другое. Диспетчер становится главным действующим лицом в управлении технологическим процессом.

Говоря о диспетчерском управлении, нельзя не затронуть проблему технологического риска. Технологические процессы в энергетике, нефтегазовой и ряде других отраслей промышленности являются потенциально опасными и при возникновении аварий приводят к человеческим жертвам, а также к значительному материальному и экологическому ущербу.

Статистика говорит, что за тридцать лет (с начала 60-х до конца 80-х годов XX века) число учтенных аварий удваивалось примерно каждые десять лет. В результате анализа большинства аварий и происшествий на всех видах транспорта, в промышленности и энергетике были получены интересные данные. В 60-х годах ошибка человека была первоначальной причиной аварий лишь в 20% случаев, тогда как к концу 80-х доля «человеческого фактора» стала приближаться к 80%.

Одна из причин этой тенденции – старый, традиционный подход к построению сложных систем управления, т. е. ориентация на применение новейших технических и технологических достижений и недооценка необходимости построения эффективного человеко-машинного интерфейса, ориентированного на человека (диспетчера). Таким образом, требование повышения надежности систем диспетчерского управления является одной из предпосылок появления нового подхода при разработке таких систем. Основа современного подхода – ориентация на оператора/диспетчера и его задачи.

Концепция **SCADA** (**S**upervisory **C**ontrol **A**nd **D**ata **A**cquisition – диспетчерское управление и сбор данных) предопределена всем ходом развития систем управления и результатами научно-технического прогресса. Применение SCADA-технологий позволяет

достичь высокого уровня автоматизации в решении задач разработки систем управления, сбора, обработки, передачи, хранения и отображения информации.

Дружественность человеко-машинного интерфейса (**HMI/MMI** – **H**umain/**M**an **M**achine **I**nterface), предоставляемого SCADA-системами, полнота и наглядность представляемой на экране информации, доступность «рычагов» управления, удобство пользования подсказками и справочной системой и т. д. повышают эффективность взаимодействия диспетчера с системой и сводят к минимуму его критические ошибки при управлении.

Следует отметить, что концепция SCADA, основу которой составляет автоматизированная разработка и управление в реальном времени, позволяет решить еще ряд задач, долгое время считавшихся неразрешимыми: сокращение сроков разработки проектов по автоматизации и прямых финансовых затрат на их разработку.

В настоящее время SCADA является основным и наиболее перспективным методом автоматизированного управления сложными динамическими системами (процессами).

Управление технологическими процессами на основе систем SCADA стало осуществляться в передовых западных странах в 80-е годы. Область их применения охватывает сложные объекты электро- и водоснабжения, химические, нефтехимические и нефтеперерабатывающие производства, железнодорожный транспорт, транспорт нефти и газа и др.

В России диспетчерское управление технологическими процессами опиралось, главным образом, на опыт оперативно-диспетчерского персонала. Переход к управлению на основе SCADA-систем стал осуществляться позднее. К трудностям освоения в России новой информационной технологии – SCADA-систем – относятся как отсутствие эксплуатационного опыта, так и недостаток информации о различных SCADA-системах. В мире насчитываются не один десяток компаний, активно занимающихся разработкой и внедрением SCADA-систем. Каждая SCADA-система – это ноу-хау компании, и поэтому информация о той или иной системе не столь обширна.

Большое значение при внедрении современных систем диспетчерского управления имеет решение следующих задач:

- выбор SCADA-системы (исходя из требований и особенностей технологического процесса);

- кадровое сопровождение.

Выбор SCADA-системы представляет собой достаточно трудную задачу, аналогичную принятию решений в условиях многокритериальности, усложненную невозможностью количественной оценки ряда критериев из-за недостатка информации.

Подготовка специалистов по разработке и эксплуатации систем управления на базе программного обеспечения SCADA осуществляется на специализированных курсах различных фирм, курсах повышения квалификации. В настоящее время в учебные планы ряда технических университетов начали вводиться дисциплины, связанные с

изучением SCADA-систем. Однако специальная литература по SCADA-системам отсутствует, имеются лишь отдельные статьи и рекламные проспекты.

Компоненты систем контроля и управления и их назначение

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

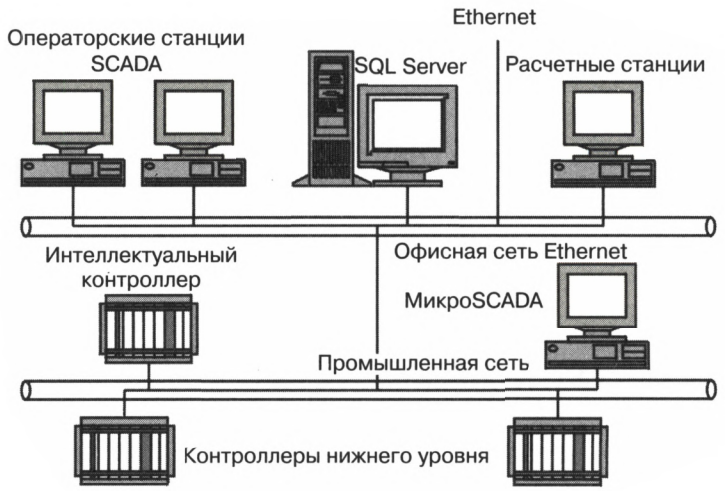


Рис. 1. *Обобщенная схема системы контроля и управления*

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

- Нижний уровень – уровень объекта (контроллерный) – включает различные датчики для сбора информации о ходе технологического процесса, электроприводы и исполнительные механизмы для реализации регулирующих и управляющих воздействий. Датчики поставляют информацию локальным программируемым логическим контроллерам (**PLC** – **P**rogramming **L**ogical **C**ontroller), которые могут выполнять следующие функции:

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

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

В качестве локальных PLC в системах контроля и управления различными технологическими процессами в настоящее время применяются контроллеры как отечественных, так и зарубежных производителей. На рынке представлены сотни типов контроллеров, способных обрабатывать от нескольких десятков до нескольких десятков тысяч переменных.

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

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

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

К этому классу инструментального ПО относятся пакеты типа ISaGRAF (CJ International France), InConrol (Wonderware, USA), Paradym 31 (Intellution, USA), имеющие открытую архитектуру.

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

  - сбор данных с локальных контроллеров;
  - обработка данных, включая масштабирование;
  - поддержание единого времени в системе;
  - синхронизация работы подсистем;
  - организация архивов по выбранным параметрам;
  - обмен информацией между локальными контроллерами и верхним уровнем;
  - работа в автономном режиме при нарушениях связи с верхним уровнем;
  - резервирование каналов передачи данных и др.

- Верхний уровень – диспетчерский пункт (ДП) – включает, прежде всего, одну или несколько станций управления, представляющих собой автоматизированное рабочее место (АРМ) диспетчера/оператора. Здесь же могут быть размещены: сервер базы данных, рабочие места (компьютеры) для специалистов и т. д. Часто в качестве рабочих станций используются ПЭВМ типа IBM PC различных конфигураций.

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

системы. SCADA – это специализированное ПО, ориентированное на обеспечение интерфейса между диспетчером и системой управления, а также коммуникацию с внешним миром.

Спектр функциональных возможностей определен самой ролью SCADA в системах управления и реализован практически во всех пакетах:

- автоматизированная разработка, дающая возможность создания ПО системы автоматизации без реального программирования;
- средства исполнения прикладных программ;
- сбор первичной информации от устройств нижнего уровня;
- обработка первичной информации;
- регистрация алармов и исторических данных;
- хранение информации с возможностью ее постобработки (как правило, реализуется через интерфейсы к наиболее популярным базам данных);
- визуализация информации в виде мнемосхем, графиков и т.п.;
- возможность работы прикладной системы с наборами параметров, рассматриваемых как «единое целое» (recipe или «установки»).

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

- Все компоненты системы управления объединены между собой каналами связи. Обеспечение взаимодействия SCADA-систем с локальными контроллерами, контроллерами верхнего уровня, офисными и промышленными сетями возложено на так называемое коммуникационное ПО. Это достаточно широкий класс программного обеспечения, выбор которого для конкретной системы управления определяется многими факторами, в том числе и типом применяемых контроллеров, и используемой SCADA-системой. Более подробная информация о коммуникационном ПО приведена в главе 2.

- Большой объем информации, непрерывно поступающий с устройств ввода/вывода систем управления, предопределяет наличие в таких системах баз данных (БД). Основная задача баз данных – своевременно обеспечить пользователя всех уровней управления требуемой информацией. Но если на верхних уровнях АСУ эта задача решена с помощью традиционных БД, то этого не скажешь об уровне АСУТП. До недавнего времени регистрация информации в реальном времени решалась на базе ПО интеллектуальных контроллеров и SCADA-систем. В последнее время появились новые возможности по обеспечению высокоскоростного хранения информации в БД. Более подробная информация по базам данных реального времени приведена в главе 6.

- Бурное развитие Интернета не могло не привлечь внимание производителей программного продукта SCADA. Возможно ли применение Интернет-технологий в системах управления технологическими процессами? Если да, то какие решения

предлагаются в настоящее время компаниями-разработчиками? Обсуждению этих вопросов посвящена глава 7.

Разработка прикладного программного обеспечения СКУ: выбор пути и инструментария

Приступая к разработке специализированного прикладного программного обеспечения (ППО) для создания системы контроля и управления, системный интегратор или конечный пользователь обычно выбирают один из следующих путей:

- программирование с использованием «традиционных» средств (традиционные языки программирования, стандартные средства отладки и пр.);
- использование существующих готовых инструментальных проблемно ориентированных средств – **COTS** (**C**ommercial **O**f **T**he **S**helf).

Для большинства выбор уже очевиден. Процесс разработки ППО важно упростить, сократить временные и прямые финансовые затраты на его разработку, минимизировать затраты труда высококлассных программистов, по возможности привлекая к разработке специалистов-технологов в области автоматизируемых процессов. При такой постановке задачи второй путь может оказаться более предпочтительным.

Для сложных распределенных систем процесс разработки собственного ППО с использованием «традиционных» средств может стать недопустимо длительным, а затраты на его разработку – неоправданно высокими. Вариант с непосредственным программированием относительно привлекателен лишь для простых систем или небольших фрагментов большой системы, для которых нет стандартных решений (не написан, например, подходящий драйвер) или они не устраивают потребителя по тем или иным причинам в принципе.

Итак, выбор пути сделан, что очень важно, но тогда следует сделать и второй шаг – «определиться» с инструментальными средствами разработки ППО.

Программные продукты класса SCADA широко представлены на мировом рынке. Это несколько десятков SCADA-систем, многие из которых нашли свое применение и в России. Наиболее популярные из них приведены ниже:

- InTouch (Wonderware) – США;
- Citect (Citect) – Австралия;
- FIX (Intellution) – США;
- Genesis (Iconics Co) – США;
- Factory Link (United States Data Co) – США;
- RealFlex (BJ Software Systems) – США;

- Sitex (Jade Software) – Великобритания;
- TraceMode (AdAstrA) – Россия;
- Cimplicity (GE Fanuc) – США;
- САРГОН (НВТ – Автоматика) – Россия.

При таком многообразии SCADA-продуктов на российском рынке естественно возникает вопрос о выборе. Выбор SCADA-системы представляет собой достаточно трудную задачу, аналогичную поиску оптимального решения в условиях многокритериальности.

Ниже приводится примерный перечень критериев оценки SCADA-систем, которые в первую очередь должны интересовать пользователя. Этот перечень не является авторским и давно уже обсуждается в специальной периодической прессе. В нем можно выделить три большие группы показателей:

- технические характеристики;
- стоимостные характеристики;
- эксплуатационные характеристики.

Технические характеристики

Программно-аппаратные платформы для SCADA-систем

Анализ перечня таких платформ необходим, поскольку от него зависит ответ на вопрос, возможна ли реализация той или иной SCADA-системы на имеющихся вычислительных средствах, а также оценка стоимости эксплуатации системы (будучи разработанной в одной операционной среде, прикладная программа может быть выполнена в любой другой, которую поддерживает выбранный SCADA-пакет). В различных SCADA-системах этот вопрос решен по разному. Так, FactoryLink имеет весьма широкий список поддерживаемых программно-аппаратных платформ (табл. 1):

Операционная система	Компьютерная платформа
DOS/MS Windows	IBM PC
OS/2	IBM PC
SCO UNIX	IBM PC
VMS	VAX
AIX	RS6000
HP-UX	HP 9000
MS Windows/NT	Системы с реализованным Windows/NT, в основном на PC-платформе

Таблица 1

В то же время в таких SCADA-системах, как RealFlex и Sitex, основу программной платформы принципиально составляет единственная операционная система реального времени QNX.

Подавляющее большинство SCADA-систем реализовано на MS Windows-платформах. Именно такие системы предлагают наиболее полные и легко наращиваемые MMI-средства. Учитывая позиции Microsoft на рынке операционных систем (ОС), следует отметить, что даже разработчики многоплатформных SCADA-систем, такие, как United States DATA Co (разработчик FactoryLink), приоритетным считают дальнейшее развитие своих SCADA-систем на платформе Windows NT. Некоторые фирмы, до сих пор поддерживающие SCADA-системы на базе операционных систем реального времени (ОСРВ), начали менять ориентацию, выбирая системы на платформе Windows NT. Все более очевидным становится применение ОСРВ во встраиваемых системах, где они действительно хороши. Таким образом, основным полем, где сегодня разворачиваются главные события глобального рынка SCADA-систем, стала MS Windows NT/2000 на фоне всё ускоряющегося сворачивания активности в области MS DOS, MS Windows 3.xx/95.

Имеющиеся средства сетевой поддержки

Одной из основных черт современного мира систем автоматизации является их высокая степень интеграции. В любой из них могут быть задействованы объекты управления, исполнительные механизмы, аппаратура, регистрирующая и обрабатывающая информацию, рабочие места операторов, серверы баз данных и т.д. Очевидно, что для эффективного функционирования в этой разнородной среде SCADA-система должна обеспечивать высокий уровень сетевого сервиса. Желательно, чтобы она поддерживала работу в стандартных сетевых средах (ARCNET, ETHERNET и т.д.) с использованием стандартных протоколов (NETBIOS, TCP/IP и др.), а также обеспечивала поддержку наиболее популярных сетевых стандартов из класса промышленных интерфейсов (PROFIBUS, CANBUS, LON, MODBUS и т.д.). Этим требованиям в той или иной степени удовлетворяют практически все рассматриваемые SCADA-системы, с тем только различием, что набор поддерживаемых сетевых интерфейсов, конечно же, разный.

Встроенные командные языки

Большинство SCADA-систем имеют встроенные языки высокого уровня, VBasic-подобные языки, позволяющие генерировать адекватную реакцию на события, связанные с изменением значения переменной, с выполнением некоторого логического условия, с нажатием комбинации клавиш, а также с выполнением некоторого фрагмента с заданной частотой относительно всего приложения или отдельного окна.

Поддерживаемые базы данных

Одной из основных задач систем диспетчерского контроля и управления является обработка информации: сбор, оперативный анализ, хранение, сжатие, пересылка и т. д. Таким образом, в рамках создаваемой системы должна функционировать база данных.

Практически все SCADA-системы, в частности Genesis, InTouch, Citect, используют ANSI SQL-синтаксис, который является независимым от типа базы данных. Таким образом,

приложения виртуально изолированы, что позволяет менять базу данных без серьезного изменения самой прикладной задачи, создавать независимые программы для анализа информации, использовать уже наработанное программное обеспечение, ориентированное на обработку данных.

Графические возможности

Для специалиста-разработчика системы автоматизации, как и для специалиста-технолога, чье рабочее место создается, очень важен графический пользовательский интерфейс. Функционально графические интерфейсы SCADA-систем весьма похожи. В каждой из них существует графический объектно-ориентированный редактор с определенным набором анимационных функций. Используемая векторная графика дает возможность осуществлять широкий набор операций над выбранным объектом, а также быстро обновлять изображение на экране, применяя средства анимации.

Крайне важен также вопрос о поддержке в рассматриваемых системах стандартных функций **GUI** (**G**raphic **U**sers **I**nterface). Поскольку большинство рассматриваемых SCADA-систем работают под управлением Windows, это и определяет тип используемого GUI.

Открытость систем

Система является открытой, если для нее определены и описаны используемые форматы данных и процедурный интерфейс, что позволяет подключить к ней «внешние» независимо разработанные компоненты.

Разработка собственных программных модулей

Перед фирмами-разработчиками систем автоматизации часто встает вопрос о создании собственных (не предусмотренных в рамках систем SCADA) программных модулей и включении их в создаваемую систему автоматизации. Поэтому вопрос об открытости системы является важной характеристикой SCADA-систем. Фактически открытость системы означает доступность спецификаций системных (в смысле SCADA) вызовов, реализующих тот или иной системный сервис. Это может быть и доступ к графическим функциям, функциям работы с базами данных и т.д.

Драйверы ввода-вывода

Современные SCADA-системы не ограничивают выбора аппаратуры нижнего уровня, так как предоставляют большой набор драйверов или серверов ввода-вывода и имеют хорошо развитые средства создания собственных программных модулей или драйверов новых устройств нижнего уровня. Сами драйверы разрабатываются с использованием стандартных языков программирования. Вопрос, однако, в том, достаточно ли спецификаций доступа к ядру системы, поставляемых фирмой-разработчиком в штатном комплекте (система Trace Mode), или для создания драйверов необходимы специальные пакеты (системы FactoryLink, InTouch). Возможно, разработку драйвера нужно заказывать у фирмы-разработчика.

Разработки третьих фирм

Многие компании занимаются разработкой драйверов, ActiveX-объектов и другого программного обеспечения для SCADA-систем. Этот факт очень важно оценивать при выборе SCADA-пакета, поскольку это расширяет область применения системы непрофессиональными программистами (нет необходимости разрабатывать программы с использованием языков C или Basic).

Стоимостные характеристики

При оценке стоимости SCADA-систем нужно учитывать следующие факторы:

- стоимость программно-аппаратной платформы;
- стоимость системы;
- стоимость освоения системы;
- стоимость сопровождения.

Эксплуатационные характеристики

Показатели этой группы критериев наиболее субъективны. Это тот самый случай, когда лучше один раз увидеть, чем семь раз услышать. К этой группе можно отнести:

- удобство интерфейса среды разработки – «Windows-подобный» интерфейс, полнота инструментария и функций системы;
- качество документации – ее полнота, уровень русификации;
- поддержка со стороны создателей – количество инсталляций, дилерская сеть, обучение, условия обновления версий и т. д.

Если предположить, что пользователь справился и с этой задачей – остановил свой выбор на конкретной SCADA-системе, то далее начинается разработка системы контроля и управления, включающая следующие этапы:

- разработку архитектуры системы автоматизации в целом. На этом этапе определяется функциональное назначение каждого узла системы автоматизации;
- решение вопросов, связанных с возможной поддержкой распределенной архитектуры, необходимостью введения узлов с «горячим резервированием» и т.п.;
- создание прикладной системы управления для каждого узла. На этом этапе специалист в области автоматизируемых процессов наполняет узлы архитектуры алгоритмами, совокупность которых позволяет решать задачи автоматизации;
- приведение в соответствие параметров прикладной системы с информацией,

которой обмениваются устройства нижнего уровня (например, программируемые логические контроллеры) с внешним миром (датчики технологических параметров, исполнительные устройства и др.);

- отладку созданной прикладной программы в режиме эмуляции.

В последующих главах на примере двух известных и хорошо зарекомендовавших себя SCADA-систем – **InTouch** и **Citect** – рассмотрены основные компоненты, функции и возможности систем диспетчерского управления и сбора данных.

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 сложными. 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. 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 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 в InTouch

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).

Каждое окно должно иметь свое имя (*Name*) для его идентификации в приложении. 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 окон, отображаемых в текущий момент. 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 всегда остается поверх всех других открытых окон. Окно закрывается после соответствующей команды пользователя.

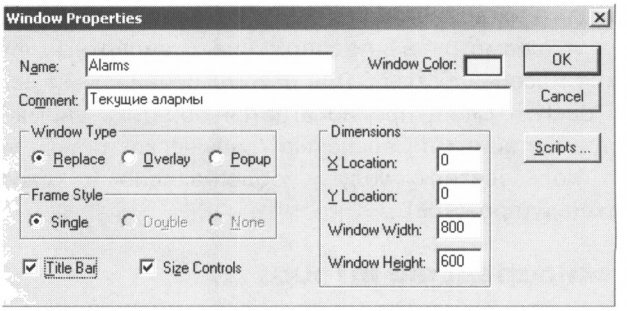


Рис. 1.1. *Диалог Window Properties (Свойства окна)*

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 без 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 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. 1.2.

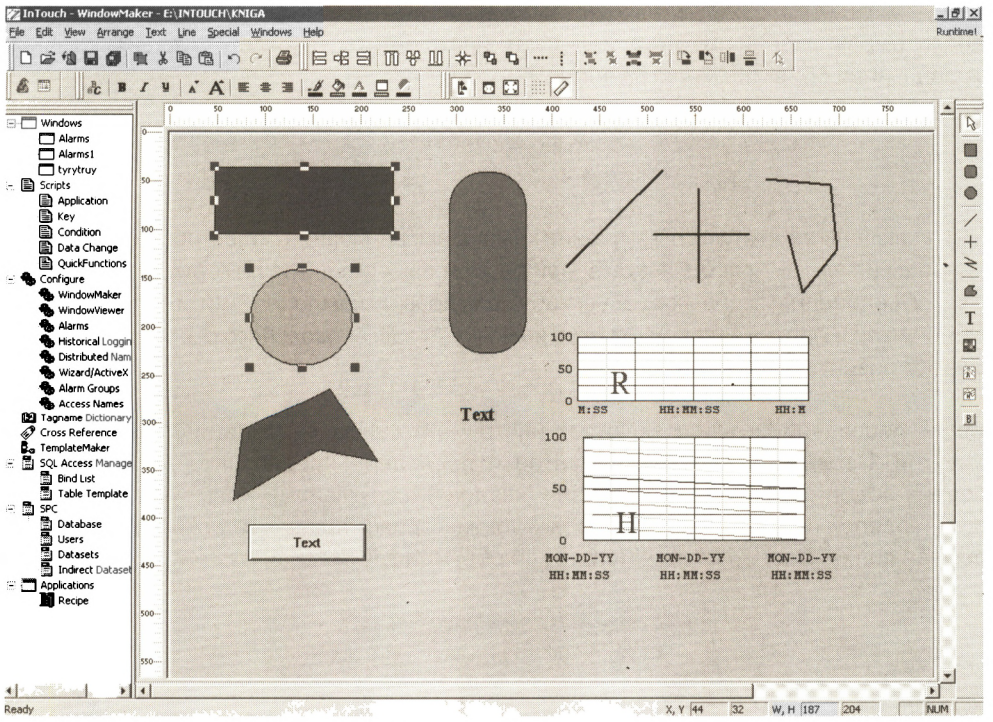


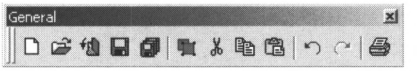
Рис. 1.2. *Интерфейс WindowMaker*

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 – меню *Application Explorer*, которое может быть выведено в интерфейс WindowMaker или закрыто нажатием соответствующей иконки инструментария.

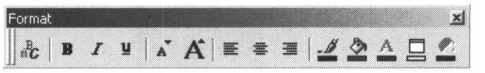
1.1.2. Инструментарий InTouch

Инструментарий InTouch представлен пятью инструментальными панелями, в которых инструменты сгруппированы по функциональному принципу.

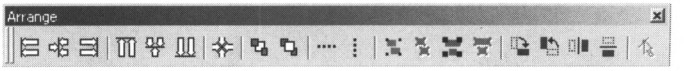
Панель **General** содержит элементы, соответствующие часто используемым командам меню *File* и *Edit*. Эти элементы известны читателю по работе в среде Windows и не требуют дополнительного пояснения.



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 90 градусов, 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. Это 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. 1.2 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. 1.2).

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 (рис. 1.3 слева), графическим объектам (в середине) и полям диалогов (справа).

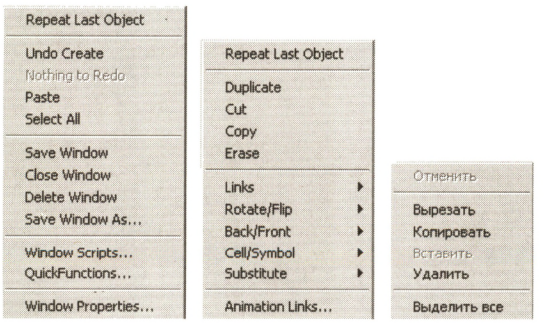


Рис. 1.3. *Меню для окна, объекта и диалога, активизируемые правой кнопкой мыши*

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, влия-

ющие на его внешний вид. Такими свойствами являются: цвет линии, цвет заполнения, высота, ширина, ориентация и т. д. 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 необходимо прежде всего вызвать на экран диалог их выбора (рис. 1.4). Это достигается командой *Special/Animation Link* или двойным щелчком левой кнопки мыши на объекте.

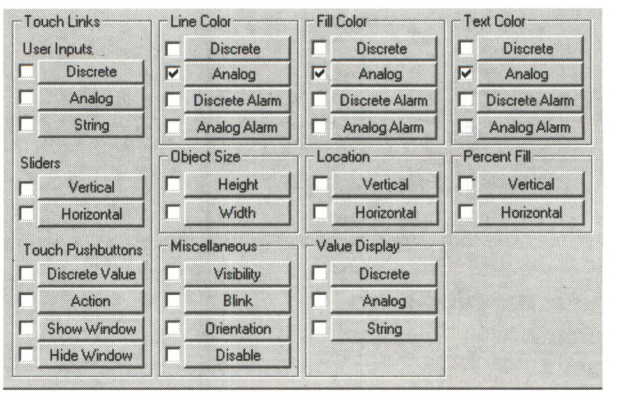


Рис. 1.4. *Диалог выбора динамических свойств объекта*

Все динамические связи можно разделить на две группы: *Touch Links* (левая колонка) и *Display Links* (три колонки справа). С помощью свойств Touch Links выполняется любой ввод в систему. Свойства Display Links осуществляют вывод информации на экран дисплея. Нажатие на любую клавишу диалога (рис. 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. 1.5 приведен диалог для определения свойств объекта (кнопки), управляющего значением дискретной переменной.

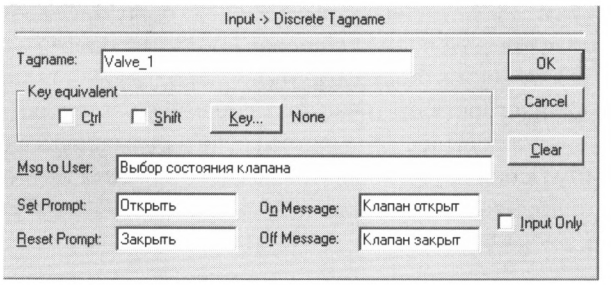


Рис. 1.5. *Диалог определения свойств кнопки*

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. 1.4). 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 его составляющие. 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. Объекты объединяются в компоненты для создания 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. 1.6).

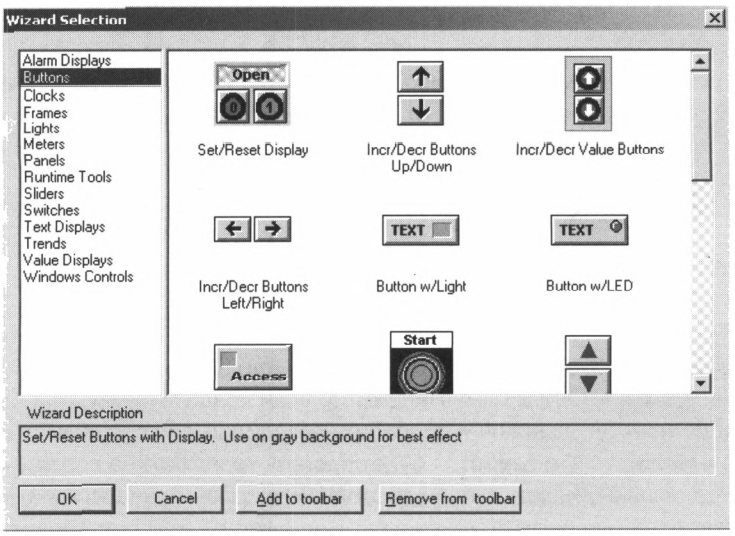


Рис. 1.6. *Диалог Wizard Selection (Выбор мастер-объекта)*

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. 1.6 – 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. 1.7).

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

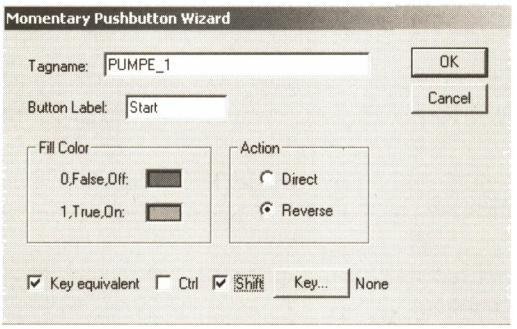


Рис. 1.7. *Диалог конфигурирования кнопки запуска*

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 переменных в реальном 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.

Подводя итог описанию графических средств пакета InTouch, следует отметить, что фирма Wonderware в этом плане предлагает потребителю хороший набор возможностей:

- богатый, традиционный для пользователей Windows инструментарий;
- меню правой кнопки мыши для окон, графических объектов и полей диалогов;
- широкий спектр динамических свойств объектов;
- огромную библиотеку мастер-объектов (Wizards).

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 редактирования графического интерфейса;
- **Cicode Editor** (редактор Cicode) – полнофункциональная интегрированная среда для создания и отладки программ на языке Cicode.

Проект Citect обычно состоит из 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 джиннов и суперджиннов.

1.2.1. Шаблоны

В Citect представлен широкий набор шаблонов практически для всех типов окон операторского интерфейса. Ниже приведено описание некоторых шаблонов, 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** – шаблон для создания меню в формате книг;
- **TabMenu** – шаблон 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, остается лишь ввести имена подключенных к ним переменных.

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 (клавиша *Prev*). Всегда под рукой у оператора находятся клавиши перехода на страницы алармов (текущие алармы, аппаратные алармы и сводка алармов).

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. 1.8).

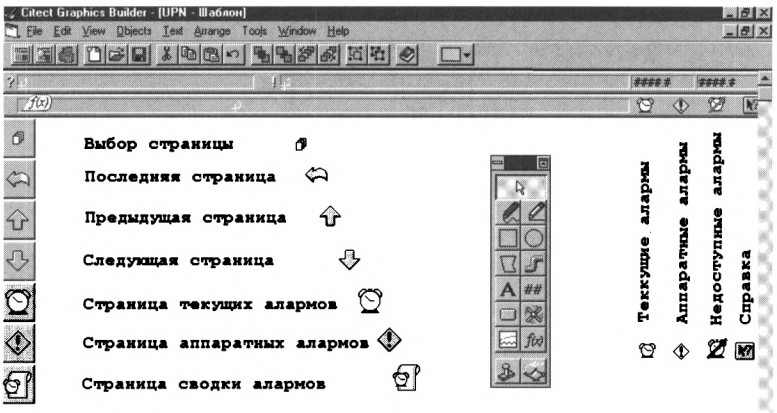


Рис. 1.8. *Шаблон страницы проекта (с описанием клавиш навигации и иконок)*

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

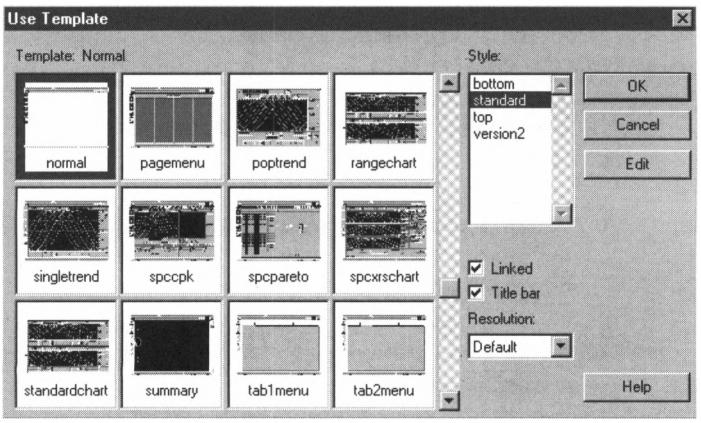


Рис. 1.9. *Диалог выбора шаблона*

По умолчанию в открывшемся диалоге (рис. 1.9) 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 диалога 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 Citect, 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 пользователя в построитель интерфейсов *Graphics Builder*.

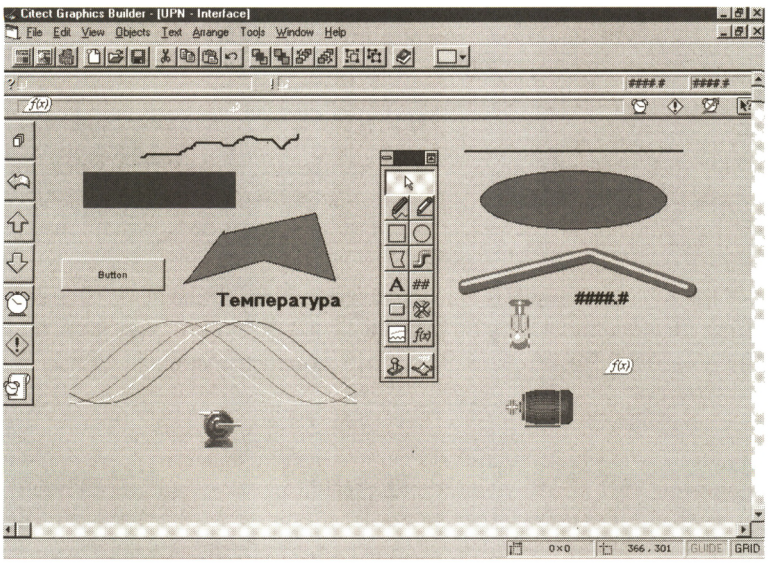


Рис. 1.10. *Страница на базе шаблона Normal с примерами объектов*

На рабочем поле окна *Graphics Builder* (рис. 1.10) представлен 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 (рис. 1.11).

произвольная линия
прямая линия
прямоугольник, квадрат
окружность, эллипс
многоугольник
трехмерный трубопровод
текст
вывод данных
кнопка
вставка объектов с заданием динамических свойств
тренд
выражение Cicode
вставка символа
вставка джинна

Рис. 1.11. *Инструментарий*

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 джинна» 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. 1.12.)

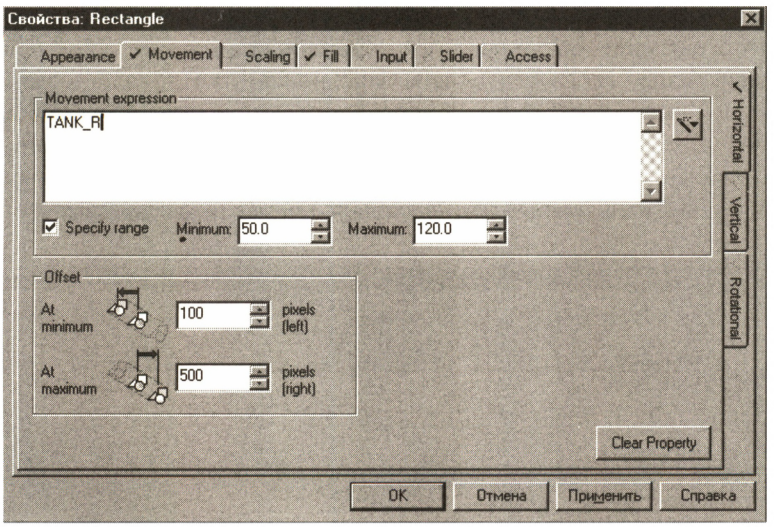


Рис. 1.12. *Диалог «Свойства объекта» с открытой закладкой Movement*

Диалог *«Свойства объекта»* содержит несколько закладок (рис. 1.12): Appearance (вид), Movement (перемещение), Scaling (масштабное изменение размеров), Fill (заполнение), Input (ввод), Slider (ползунок), Access (доступ). Щелчок мыши по 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. 1.12) 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 соответственно увеличивается/уменьшается. В диалоге *Scaling* предлагается определить выражение, вызывающее изменение ширины объекта, его минимальное и максимальное значение, 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. 1.13 приведен диалог *Fill/Level* (уровень заполнения), где предлагается определить переменную или выражение, которые определяют изменение уровня в объекте (поле *Level Expression*), минимальное и максимальное значения уровня (Minimum, Maximum), процент закрашивания объекта при минимальном и максимальном значениях уровня (поле Level), а также направление закрашивания объекта (кнопки Fill Direction).

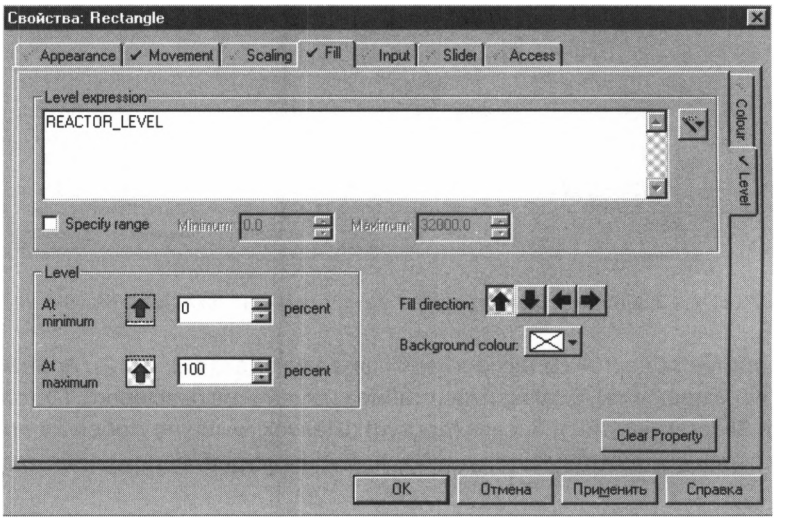


Рис. 1.13. *Диалог «Свойства объекта» с открытой закладкой Fill*

Закладка *Input* предоставляет разработчику возможность связать с объектом некоторую команду (поле *Up Command*, рис. 1.13), 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 определяются зоны и объекты, доступные каждому из пользователей. Например, доступ к таким объектам, как регулятор, предоставляется не всем операторам, и 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. 1.12) имеет три боковые закладки, связанные с типом перемещения: горизонтальное (Horizontal), вертикальное (Vertical) и вращательное (Rotational).

В диалоге *Fill* (рис. 1.13) представлены 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 требуется вводить имена переменных, используемых в проекте, и функции Cicode. С одной стороны, это занимает много времени, с другой – повышается вероятность ошибки 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. 1.14).

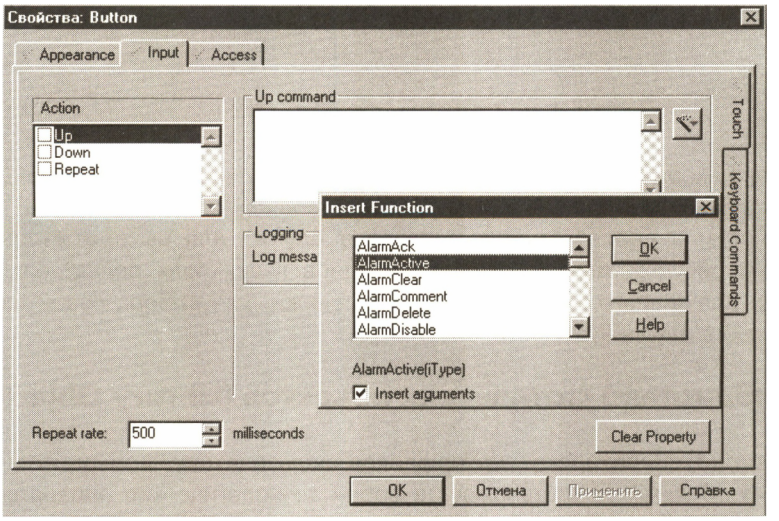


Рис. 1.14. *Диалог «Свойства объекта» на закладке Input со списком Cicode-функций*

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

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

- существенно возрастает скорость обучения новых пользователей;
- упрощается работа по созданию графического интерфейса;
- сокращаются сроки разработки проекта.

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. Общий вид редактора Bitmap Editor с фрагментом нарисованного объекта показан на рис. 1.15.

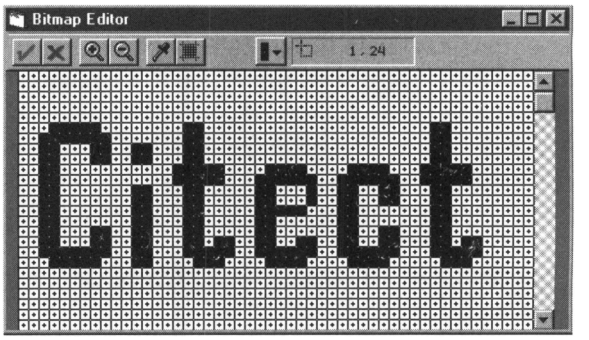


Рис. 1.15. *Редактор Bitmap Editor*

Когда курсор находится на рабочем поле редактора, он имеет форму карандаша. Щелчок левой кнопки мыши по ячейке вызывает ее заполнение предварительно выбранным цветом. Кнопки в верхней части редактора служат для выхода из редактора с сохранением рисунка, для изменения масштаба ячеек поля редактора (увеличить, уменьшить), для точного определения цвета ячейки, для выбора размеров поля под рисунок, а также для вызова цветовой палитры.

1.2.4. Библиотека статических объектов (Library Objects)

При разработке операторских интерфейсов пользователю приходится применять 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. 1.16 представлен набор объектов раздела «Емкости».

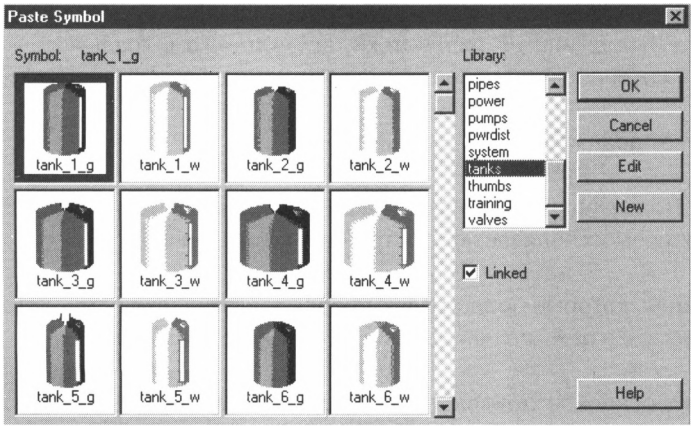


*Рис. **1.16.** Раздел «Емкости» библиотеки объектов*

Теперь нет необходимости рисовать объект и терять драгоценное время, если подобный объект есть в библиотеке. Достаточно открыть библиотеку объектов щелчком по иконке инструментария, выбрать раздел, затем – объект и вставлять его в любые окна приложения. Операция вставки готового объекта занимает всего несколько секунд.

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. При модификации графического объекта в библиотеке автоматически меняется его образ во всех окнах, где он используется.

1.2.5. Джинны и суперджинны (Genies и Super Genies)

Каждый 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 джинном. Можно объединить несколько связанных задачей объектов в группу, предварительно придав этим объектам соответствующие свойства, а затем сохранить эту группу в библиотеке джиннов, которая устроена аналогично библиотеке объектов. Джинн может управляться как единый объект (его можно копировать, перемещать, масштабировать и т.д.), при этом обрабатываются все составляющие джинна.

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

Citect предлагает два типа сложных объектов:

- **джинны**, которые размещаются на графической странице при проектировании системы, причем их количество на странице не ограничено;

- **суперджинны**, которые представляют собой динамические страницы, активизируемые в режиме исполнения для ввода/вывода данных.

Таким образом, основное отличие этих механизмов в том, что джинн объединяет несколько объектов и привязан к странице, а суперджинн является отдельной страницей.

В каких случаях может быть полезен суперджинн?

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 суперджинн (выпадающая страница).

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

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

Пакет Citect поставляется с библиотекой джиннов и суперджиннов. Вызов библиотеки производится автоматически при выборе инструмента *Paste Genie* (вставка джинна).

На рис. 1.17 приведен раздел «Моторы» библиотеки джиннов.

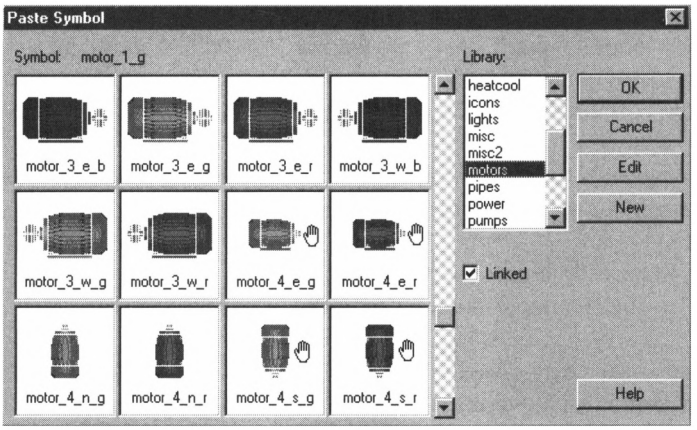


Рис. 1.17. *Раздел «Моторы» библиотеки джиннов*

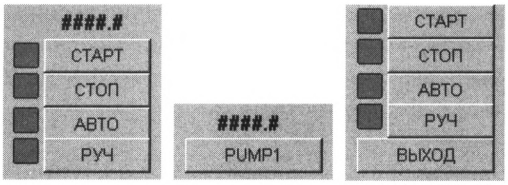
Часто суперджинны и джинны используются вместе. Это достигается привязкой джинна к суперджинну, когда одна из функций джинна активизирует суперджинн (выпадающую страницу). В библиотеке джиннов Citect некоторые джинны уже связаны с суперджинами (джинны с символом руки).

Механизм работы джинна, связанного с суперджинном, будет более понятен, если внимательно прочитать следующие несколько абзацев. Следует еще раз подчеркнуть, что речь пойдет только о механизме «срабатывания», а не о методике создания джиннов и суперджиннов.

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 джинн, реализующий все эти функции. Примерный вид этого джинна представлен на рис. 1.18а.

На мнемосхеме – несколько насосов, и для каждого нужен свой джинн. Citect допускает любое количество джиннов на странице, но она будет перегружена информацией, которая не нужна оператору постоянно.



а) ***б)*** ***в)***

Рис. 1.18. *К описанию механизма суперджинна*

Предлагается второе решение этой задачи – создать джинн и суперджинн. Постоянно на мнемосхеме процесса присутствуют джинны для управления насосами, один из которых представлен на рис. 1.18б. Но в этом случае они намного компактнее и не перегружают интерфейс. При определении свойств кнопки PUMP1 (насос1) джинна на закладке INPUT (см. рис. 1.14) 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 суперджинна;
- sTag1..8 – имена переменных, связанных с суперджинном.

Джинн, связанный с суперджинном (рис. 1.18в), заранее создан и сохранен в библиотеке джиннов и суперджиннов. При определении свойств компонентов (кнопок) этого суперджинна должны быть использованы заменяемые имена с синтаксисом ?type number?:

- type – тип переменной (string, integer, real или digital);
- number – позиция имени переменной (1-8) в списке функции AssPopUp() джинна, который вызывает страницу суперджинна.

Применительно к рассматриваемому примеру функция (команда), вызывающая суперджинн, может иметь вид AssPopUp(«SGenie», «%PUMP%», «%STATUS%»), а заменяемые имена переменных суперджинна – ?digital 1? и ?digital 2?.

Кнопки суперджинна SGenie (рис. 1.18в) 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 джинна 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? суперджинна, что вызовет появление выпадающей страницы с пультом управления насосом PUMP1 (рис. 1.18в). После выполнения действий по управлению насосом выпадающая страница может быть закрыта щелчком по кнопке ВЫХОД (функция WinFree( ) закрывает страницу).

В результате применения суперджиннов 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.

- Citect – компания с большим опытом разработки проектов. Взгляд Citect – 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. Знакомый многим 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 появится соответствующая информация с целью модификации неправильных параметров.

- Citect предлагает библиотеки символов, джиннов и суперджиннов. 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 только на этапе компиляции проекта.

- 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 де-факто 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 (**O**LE for **P**rocess **C**ontrol) заключается в определении механизма доступа к данным с любого устройства системы управления. 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 контроля и управления, работающие в распределенной гетерогенной (неоднородной) 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 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:

- **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 ПО

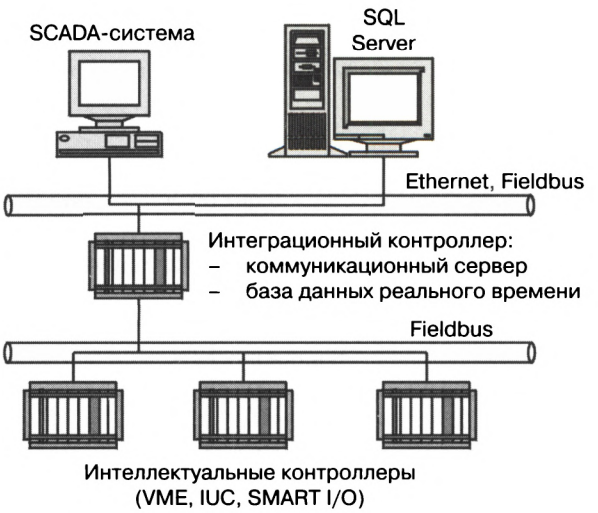


Рис. 2.1. *Типовая архитектура системы управления*

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. 2.1).

Коммуникационное ПО 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 компоненты:

- статическую библиотеку, используемую с традиционными языками программирования, такими, как C, C++, Pascal;
- динамическую библиотеку (DLL), применяемую со всеми Windows языками программирования (Visual Basic, Visual C/C++, Borland C/C++, Delphi, LabWindows CVI, LabView);
- DDE-сервер, имеющий 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 показаны программные интерфейсы для Windows-приложений (в том числе и SCADA-систем) и спектр широко распространенных промышленных протоколов.

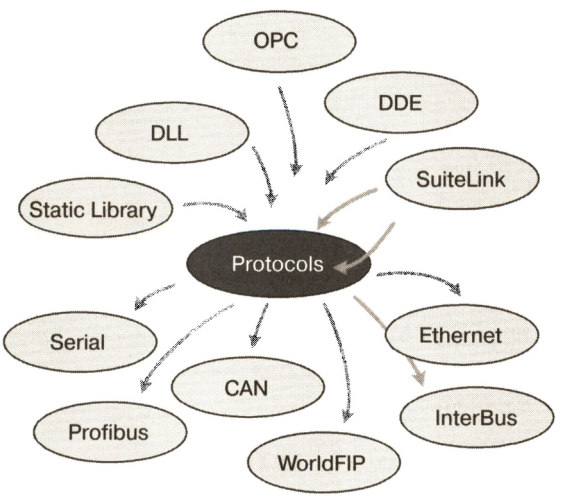


Рис. 2.2. *Набор интерфейсов для SCADA-систем и спектр поддерживаемых протоколов*

Использование этих протоколов позволяет организовать взаимодействие с контроллерами, устройствами, объединенными промышленными (fieldbuses) и обычными сетями. Предлагаемая схема решения позволяет конечному пользователю, системному интегратору единообразным способом организовать взаимодействие между ПО верхнего уровня и платами, специфичными для каждого типа промышленных сетей.

DDE-, OPC-компоненты являются серверами по отношению к SCADA-системам. По отношению к ПО нижнего уровня (fieldbus) возможна организация Master/Slave и Client/Server. Внешние устройства способны посылать и принимать данные через плату. Когда вставляемая в персональный компьютер плата является Master/Client, то именно плата с поддерживаемым ПО является инициатором опроса промышленных устройств. В случае применения плат типа Slave/Server они реагируют на запросы внешних устройств.

На некоторых вставных платах имеется разделяемая область памяти. Эта память доступна как приложению в ПК, так и встраиваемому ПО.

На рис. 2.3 представлена 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.

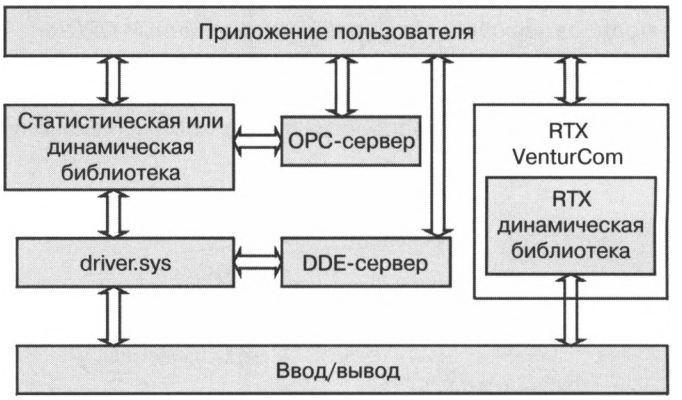


Рис. 2.3. *Схема организации коммуникационного ПО для Windows NT*

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 перемен-

ной, 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 (**D**ynamic **D**ata **E**xchange – 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. Созрела необходимость в появлении более совершенного технологичного протокола. 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** (**V**alue, **T**ime, **Q**uality – 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** (**D**istributed **C**omponent **O**bject **M**odel) 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. 2.4).

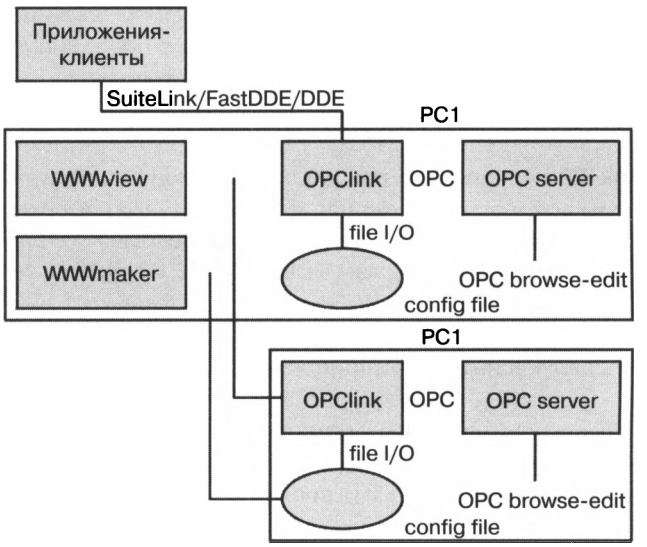


Рис. 2.4. *Использование SuiteLink-протокола в SCADA-системах*

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. 2.5), 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).

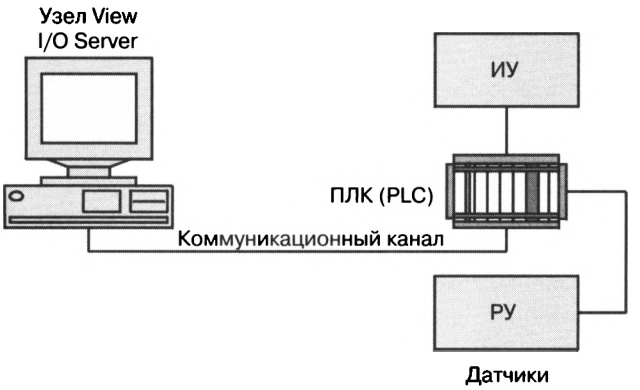


Рис. 2.5. *Обмен данными между InTouch-приложением и технологическим процессом*

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. 2.6).

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

Рис. 2.6. *Схема обмена информацией InTouch-приложения с ПЛК*

Для организации обмена с приложением определяются каналы обмена или каналы доступа. Их параметры:

- имя узла (Node Name);
- имя приложения (Application Name);
- имя группы данных, или топика (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. 2.7).

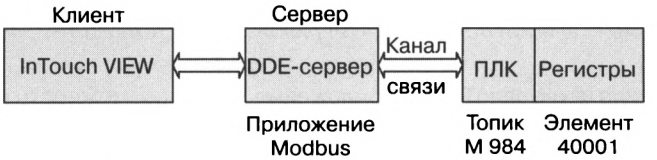


Рис. 2.7. *Обмен информацией на примере Modbus-сервера*

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. 2.6 и 2.7, 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

Приложения InTouch могут взаимодействовать не только между собой, но и с другими Windows-приложениями, например с 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. Такой 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 (Topic Name) 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).

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

Таблица 2.1

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, следует подчеркнуть, что информация по доступу к данным устройств ввода/вывода или других приложений должна храниться в приложении (в словаре переменных). И разработчику InTouch-приложения 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 ни 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 используются 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.

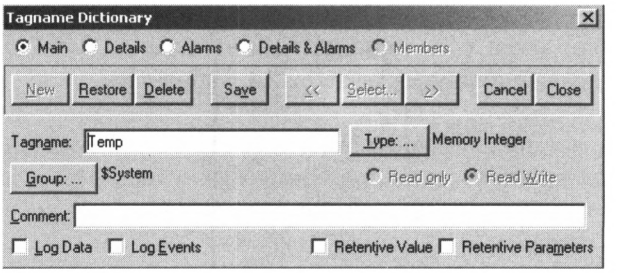


Рис. 2.8. *Диалог Tagname Dictionary (Словарь переменных)*

Определение новой переменной в базе данных InTouch, как и просмотр и модификация атрибутов уже существующих переменных, производится в диалоге *Tagname Dictionary* (рис. 2.8). 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*.

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.

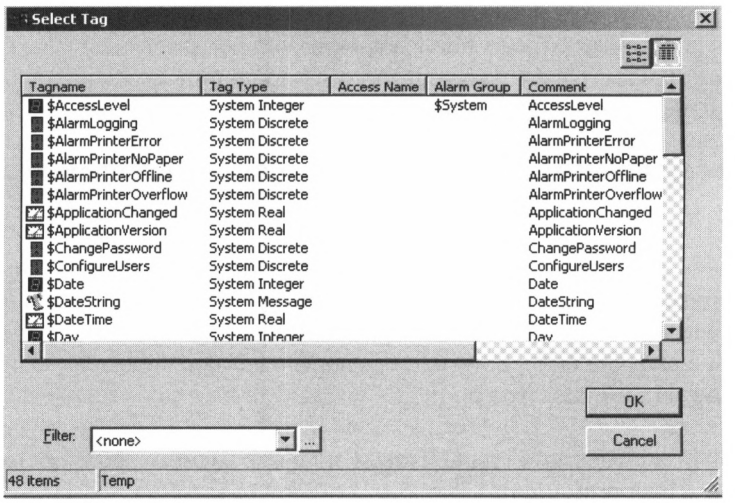


Рис. 2.9. *Диалог Select Tag (Выбор переменной)*

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. 2.9. Для каждой переменной в этом диалоге приведена следующая информация: имя переменной, ее тип, имя доступа, группа аларма и комментарий.

Группа алармов (Alarm group, рис. 2.9) для переменной определяется в диалоге, вызываемом нажатием кнопки *Group* диалога *Tagname Dictionary*. Ответы на многие вопросы, связанные с разработкой подсистемы алармов, можно найти в следующей главе.

Выбор типа переменной осуществляется в диалоге *Tag Types* (Тип переменной, рис. 2.10), вызываемом на экран нажатием кнопки *Type* диалога *Tagname Dictionary*. В этом диалоге представлен полный список основных типов переменных InTouch. Выбор завершается отметкой соответствующей опции и щелчком по кнопке *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.

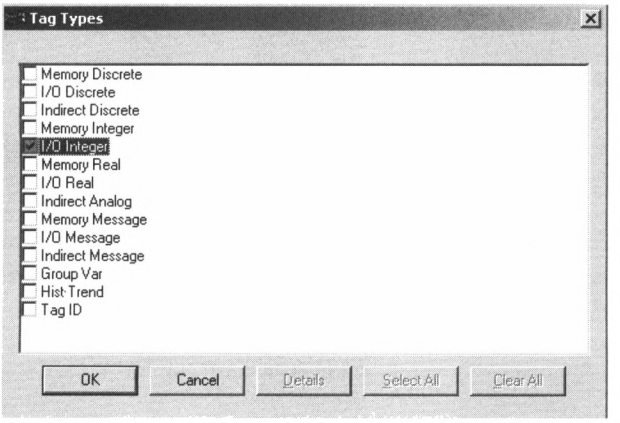


Рис. 2.10. *Диалог Tag Types (Тип переменной)*

На рис. 2.11 представлен диалог подробного описания вещественной переменной типа *I/O Integer*.

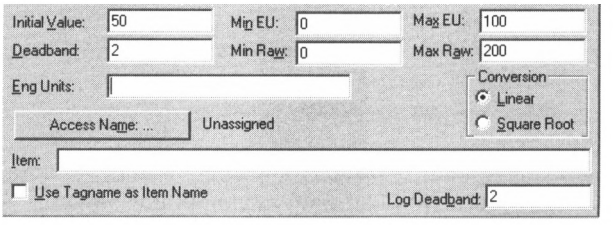


Рис. 2.11. *Диалог подробного описания переменной типа I/O*

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 (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. 2.11).

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

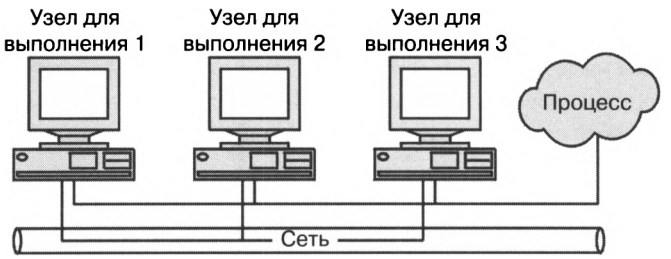


Рис. 2.12. *Сеть View-узлов с собственными серверами ввода/вывода*

Локальные адреса используются в том случае, когда View-узлы имеют свои серверы ввода/вывода. На рис. 2.12 узлы исполнения (View-узлы), каждый со своей копией одного и того же приложения, ссылаются на свои собственные источники данных ввода/вывода (серверы 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. 2.11) вызывает на экран одноименный диалог.

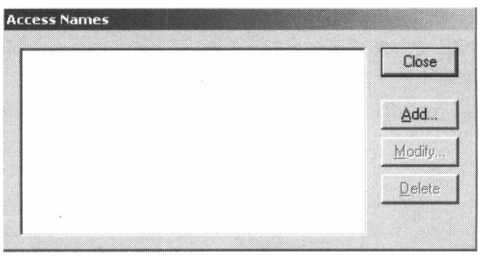


Рис. 2.13. *Диалог Access Names (Имена доступа)*

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. 2.14).

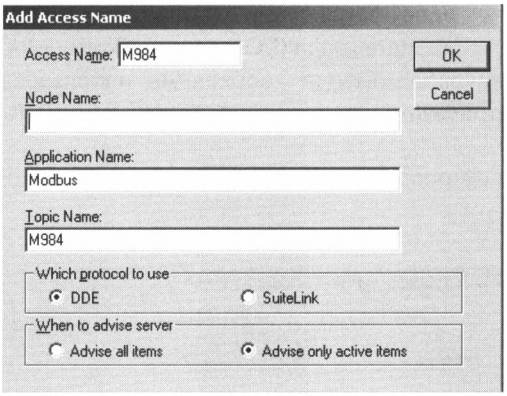


Рис. 2.14. *Диалог определения нового канала доступа (локальный адрес)*

Диалог определения канала доступа заполнен в соответствии с примером, рассмотренным на рис. 2.7. В качестве имени (канала) доступа (Access Names) рекомендуется выбирать имя группы (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. 2.15).

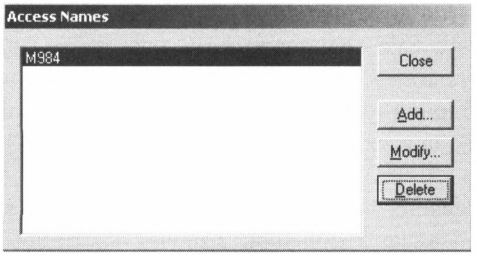


Рис. 2.15. *Диалог Access Names с определенным именем доступа*

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. 2.16).

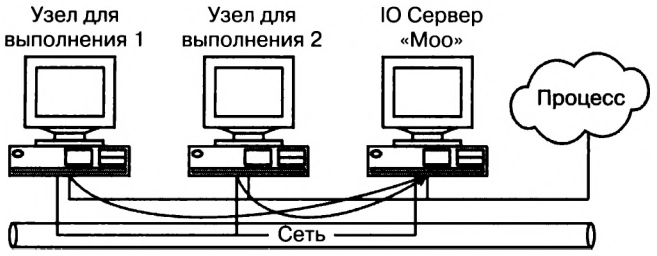


Рис. 2.16. *Архитектура с двумя View-узлами и сервером ввода/вывода*

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. 2.17.

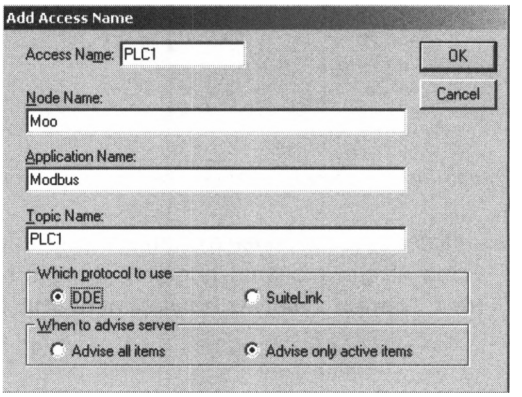


Рис. 2.17. *Диалог определения нового канала доступа (глобальный адрес)*

При выборе имени доступа действует то же правило, что и при локальной адресации: рекомендуется, чтобы это имя совпадало с именем группы данных (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: 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.

  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 то же 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. 2.18).

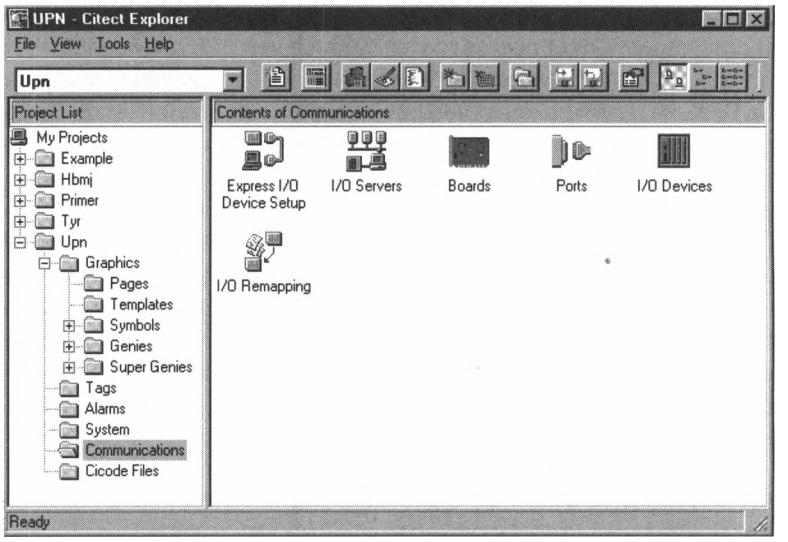
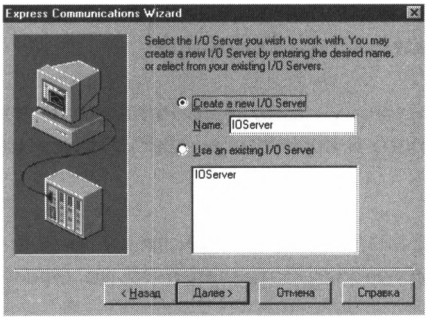


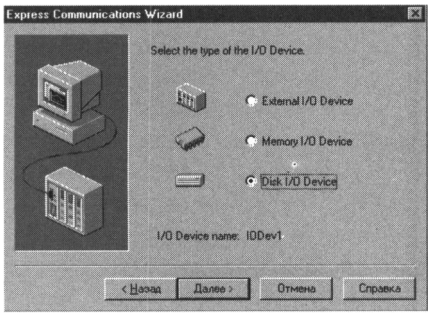
Рис. 2.18. *Доступ к мастеру коммуникаций из Citect Explorer*

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.

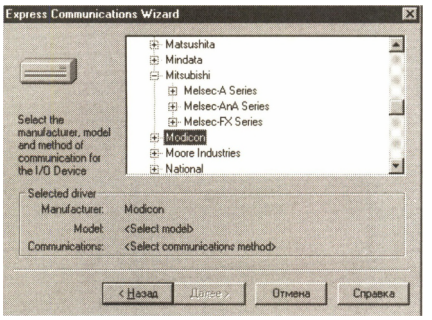
Последовательное нажатие клавиши *Next* (далее) открывает перед разработчиком новые диалоги, предлагая ввести необходимую информацию по установке связи между Citect и устройством ввода/вывода.



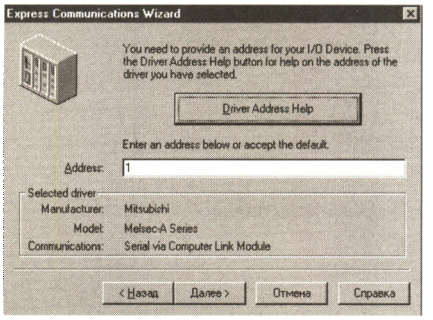
Citect предоставляет возможность пользователю разрабатывать и отлаживать проект без необходимости физического подключения к реальному устройству ввода/вывода. Просто при конфигурировании устройства ввода/вывода его можно определить как внутреннее (Memory I/O Device) или как диск (Disk I/O Device).

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. 2.18), 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. 2.19.

- В диалоге **Server** (сервер) для определения сервера задают его имя в поле Server Name. При наличии двух серверов (дублирование) каждый 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).

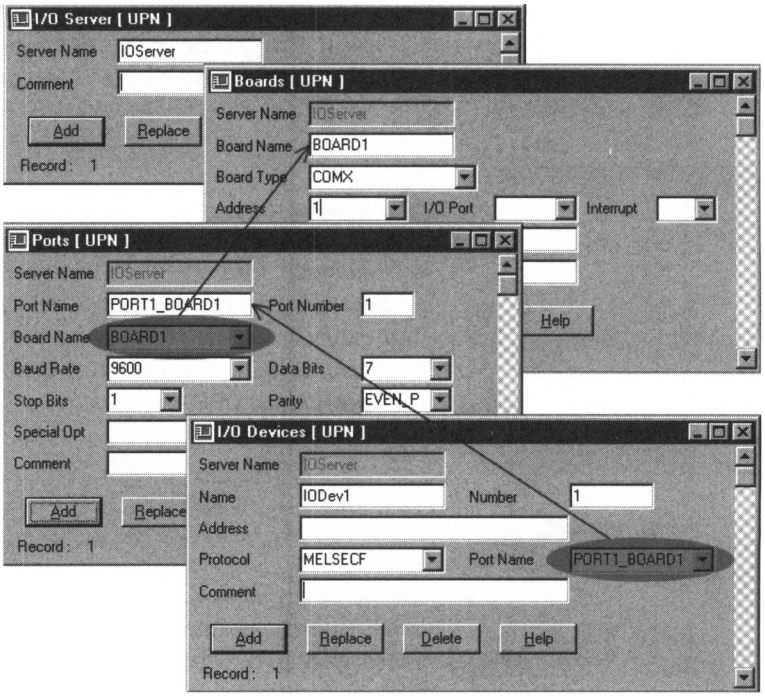


Рис. 2.19. *Диалоги конфигурирования параметров связи*

- 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 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. 2.20).

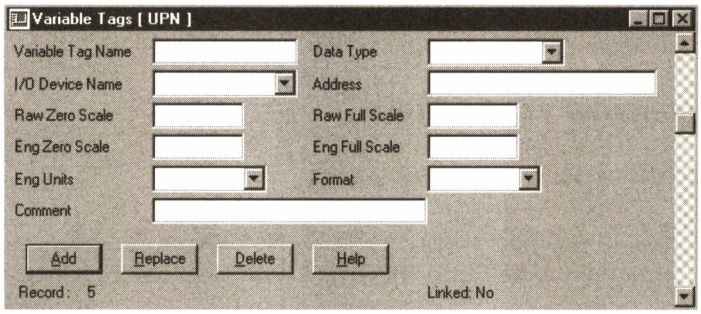


Рис. 2.20. *Диалог Variable Tags (Переменная)*

Для каждой 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, однотипна, при большом количестве переменных процесс будет достаточно трудоемким.

Variables_DBF

	A	B	C	D
1	NAME	TYPE	UNIT	ADDR
2	TEST	DIGITAL	IODev2	DO
3	MALT_LEVEL	INT	IODev1	I2
4	MILL_SPEED	INT	IODev1	I3
5	HW_TEMP	INT	IODev1	I4
6	HOPS_LEVEL	INT	IODev1	I5
7	KETTLE_TEMP	INT	IODev1	I6
8	MILL_STAT	DIGITAL	IODev1	D1
9	MT_STAT	DIGITAL	IODev1	D2
10	EXT_STAT	DIGITAL	IODev1	D3
11	WP_STAT	DIGITAL	IODev1	D4
12	MALT_VALVE	DIGITAL	IODev1	D5
13	HW_VALVE	DIGITAL	IODev1	D6
14	MASH_VALVE	DIGITAL	IODev1	D7
15	MASH_PUMP	DIGITAL	IODev1	D8
16	HOPS_VALVE	DIGITAL	IODev1	D9
17	BW_VALVE	DIGITAL	IODev1	D10
18				
19				
20				

Лист1 / Лист2 / Лист3

Рис. 2.21. *Фрагмент базы данных в таблице Excel*

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. 2.21.

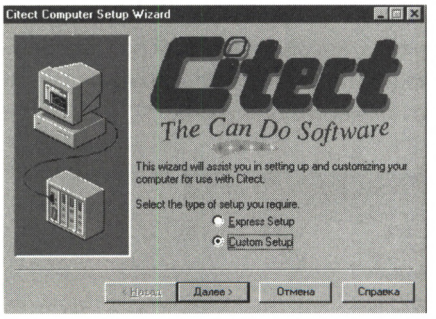
2.5. Подключение узлов Citect

2.5.1. Архитектура клиент-сервер

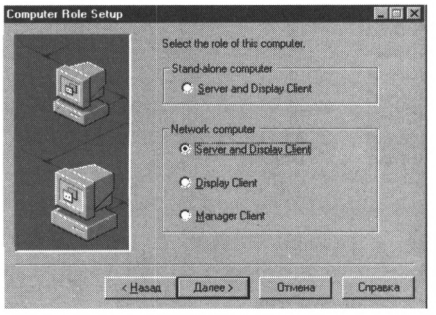
Сетевые средства Citect построены на базе NetBIOS и поддерживаются такими сетевыми протоколами, как NetBEUI, IPX/SPX, TCP/IP. С другой стороны, Citect поддерживает все сетевые протоколы, совместимые с NetBIOS, что существенно расширяет спектр сетей, с которыми может взаимодействовать Citect. К таким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

2.5.2. Конфигурирование Citect-компьютеров в сети

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

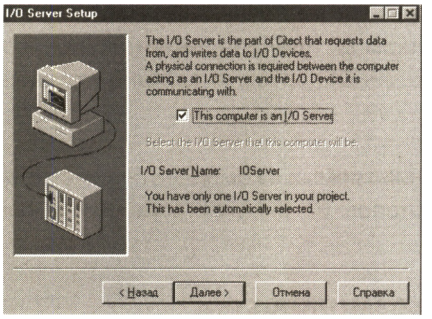


Первый диалог предлагает выбрать режим работы программы. Работа Computer Setup Wizard может производиться как в экспресс-режиме (рекомендуемые параметры), так и в режиме выборочной установки (пользовательские параметры). Последовательное нажатие клавиши *Next* (далее) открывает перед разработчиком новые диалоги, предлагая ввести необходимую информацию по конфигурированию Citect-компьютера в сетевой архитектуре.



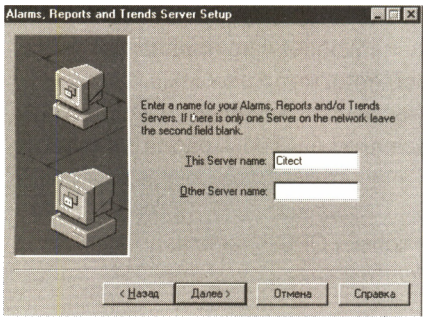
В этом окне определяется роль данного узла в Citect-системе: компьютер, выполняющий функции клиента и сервера, только клиента или мониторинговые функции. Здесь же надо определить, сетевым или автономным является компьютер.

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.

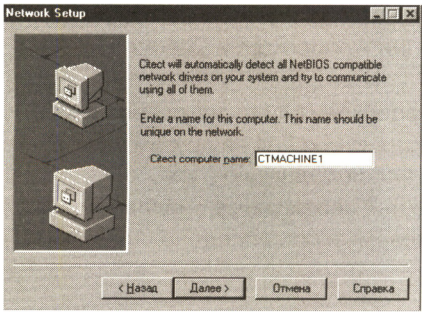


Если в предыдущем диалоге выбрать опцию *Server and Display Client* в сетевом применении, то далее будут открываться диалоги с предложениями определить этот компьютер как сервер ввода/вывода, сервер алармов, трендов, отчетов.

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 компании Wonderware и Citect 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 Citect поставляет встроенные драйверы, сводя 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 Citect:

- 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. 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. Принял ли оператор сообщение об аларме? Чтобы снять эти сомнения, в системах управления принято различать неподтвержденные и подтвержденные алармы. Аларм считается подтвержденным после того, как оператор отреагировал на сообщение об аларме. До этого аларм считается неподтвержденным.

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 в худшем – 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 разные. С другой стороны, когда речь идет о типах алармов, то все SCADA-системы поддерживают **дискретные** и **аналоговые** 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 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 задаются в нескольких комбинациях:

- **High** и **High High** (верхний и выше верхнего);

- **Low** и **Low Low** (нижний и ниже нижнего);
- **Deviation** (отклонение от нормы);
- **ROC** – **R**ate **o**f **C**hange (скорость изменения).

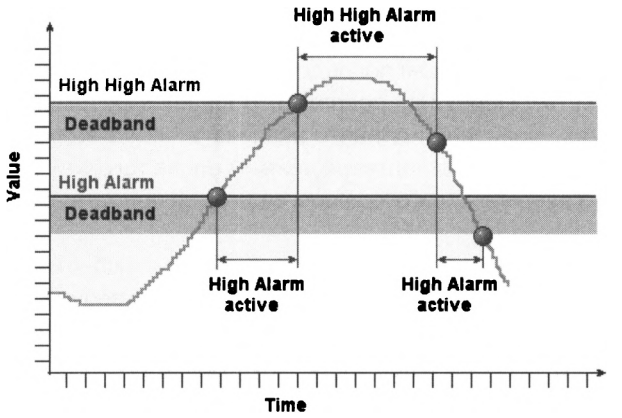


Рис. 3.1. *Графическая интерпретация алармов типа Hi и HiHi*

На рис. 3.1 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. 3.2), 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).

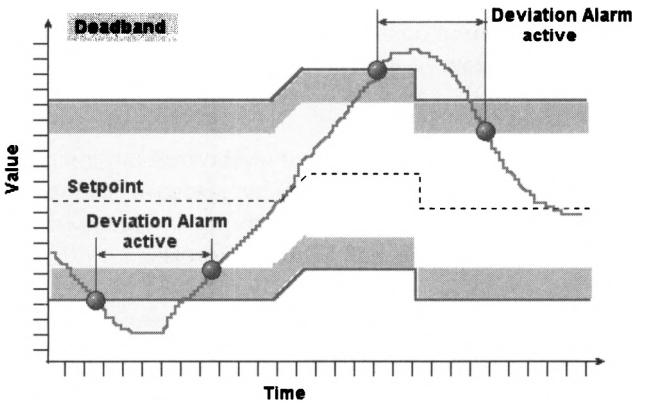


Рис. 3.2. *Графическая интерпретация алармов типа Deviation*

Аларм сработает при выходе значения переменной за границу предельно допустимого отклонения.

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 имеются 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

Каждая переменная связывается с какой-либо группой алармов. Все эти группы определяются пользователем и могут быть объединены в иерархическую структуру. 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.

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. Если пользователь не определил группу алармов для конкретной переменной, то она по умолчанию связывается с корневой группой $System. А все определяемые группы алармов становятся потомками этой группы.

Иерархическая древовидная структура может иметь до восьми уровней, при этом каждая входящая в дерево группа может иметь до 16 подгруп (рис. 3.3).

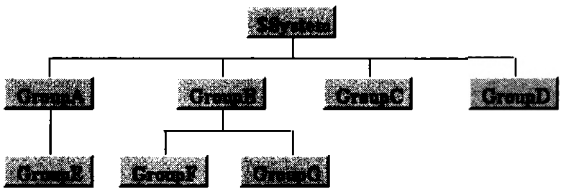


Рис. 3.3. *Иерархическая древовидная структура групп алармов*

Для создания таких групп в меню окна WindowMaker предусмотрена команда *Special/Alarm Groups* (Группы алармов), вызывающая появление диалога *Alarm Groups* (рис. 3.4). При определении переменных в словаре *Tagname Dictionary* нажатие кнопки *Group* (см. рис. 2.8) также выводит на экран этот диалог.

Рис. 3.4. *Диалог Alarm Groups (Группы алармов)*

Воспользовавшись кнопкой *Add*, можно добавить группу алармов, а также сформировать древовидную структуру системы алармов, определяя родительские группы и группы-потомки. При этом открывается диалог (рис. 3.5) *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.

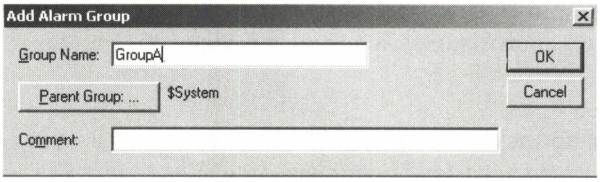


Рис. 3.5. *Диалог Add Alarm Group (Добавить группу алармов)*

На рис. 3.6 диалог *Alarm Groups* (Группы алармов) заполнен в соответствии с древовидной структурой групп алармов, представленной на рис. 3.3.

Рис. 3.6. *Диалог Alarm Groups*

3.2.4. Определение условий аларма для переменной

Условия возникновения состояния аларма 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. 2.11). Для дискретной переменной этот диалог имеет следующий вид:

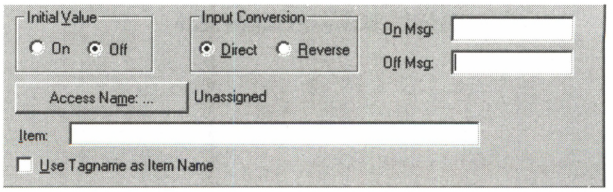


Рис. 3.7. *Диалог подробного описания дискретной переменной*

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 алармах. В случае возврата переменной в нормальное состояние запись о ней исчезает из текущей аварийной сводки.

С помощью объекта «Архивная сводка алармов» на дисплей выводятся все данные об алармах и событиях, включая количество уже произошедших алармов данного типа, время подтверждения, время возврата переменных 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 *Standard Alarm Displays* (стандартная система алармов). Осталось щелкнуть по *Ok* и вставить объект вывода аварийной информации в окно (рис. 3.8).

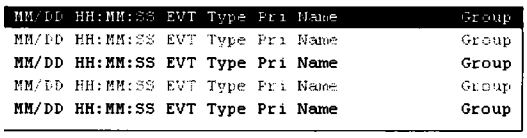


Рис. 3.8. *Стандартный объект вывода аварийной информации*

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. 3.9).

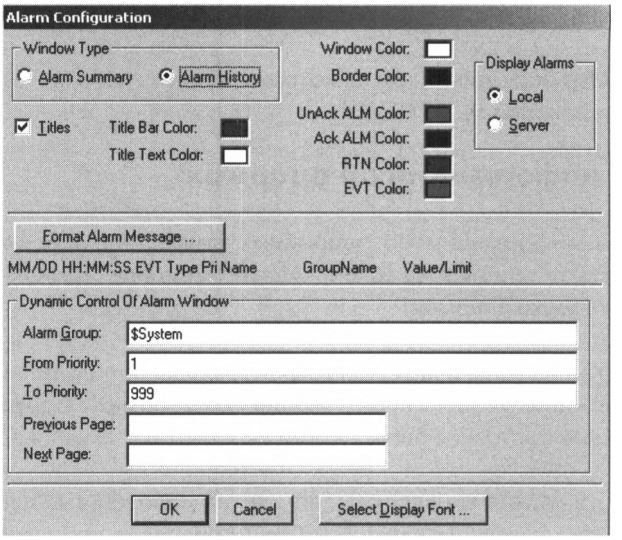


Рис. 3.9. *Диалог Alarm Configuration*

Быстрый доступ к этому диалогу можно получить, воспользовавшись меню правой кнопки мыши с последующим щелчком на строке *Properties*. 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 необходимо воспользоваться кнопкой *Select Display Font*.

Нажатие кнопки *Format Alarm Message* (форматирование аварийного сообщения) выводит на экран одноименный диалог (рис. 3.10), где определяется информация, включаемая в аварийное сообщение.

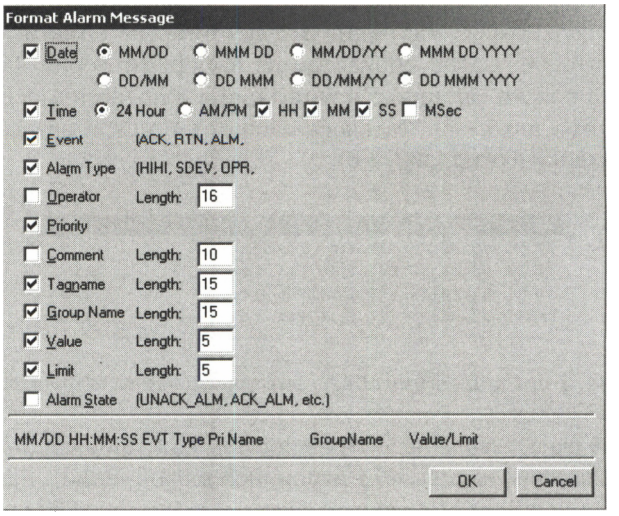


Рис. 3.10. *Диалог Format Alarm Message*

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. 3.8.

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. 3.11.

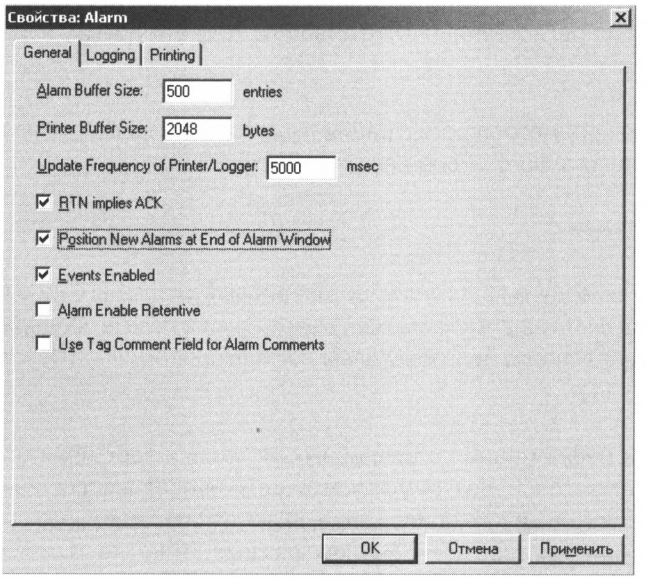


Рис. 3.11. *Диалог Alarm Properties (Свойства алармов)*

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. 3.11).

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. 3.11).

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 использовать эту систему и для сетевых приложений.

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. 3.9) 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. 3.12). Этот диалог вызывается командой *Special/Configure/WindowViewer*. Для быстрого вывода этого диалога надо дважды щелкнуть на строке *WindowViewer* группы *Configure* окна *Application Explorer*.

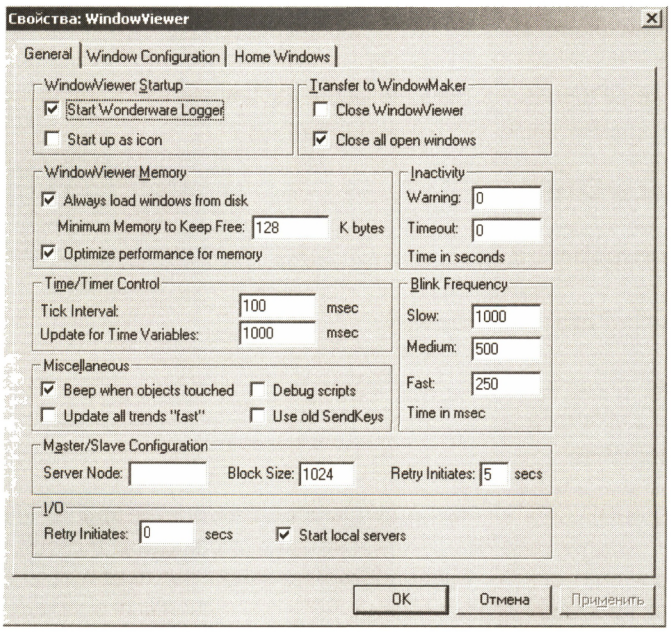


Рис. 3.12. *Диалог «Свойства WindowViewer»*

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 InTouch-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.

3.3. Алармы в Citect

3.3.1. Типы алармов

Citect поддерживает два типа алармов:

- **аппаратные**;
- **конфигурируемые**.

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 на дисплей оператора. Это свойство Citect является встроенным и не нуждается в предварительной настройке (конфигурировании). Аппаратные алармы отображаются на специальной странице (Hardware Alarm Page).

Алармы, вызываемые выходом значений 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);
- **аналоговые** (analog);
- **с метками времени** (time stamped);
- **составные** (advanced).

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.

Срабатывание аналоговых алармов происходит в результате выхода значений переменной за заданные 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);
- **ROC** – **R**ate **O**f **C**hange (скорость изменения).

Алармы с меткой времени подобны дискретным алармам – срабатывают 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. 3.13 приведен интерфейс Citect Explorer с открытой папкой Alarms.

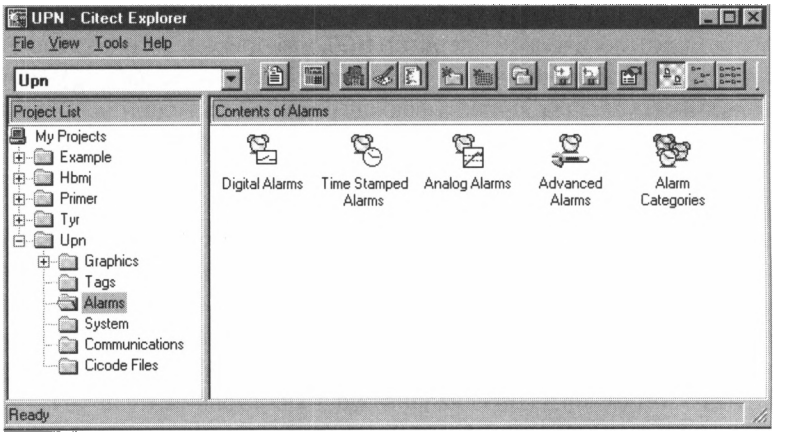


Рис. 3.13. *Интерфейс Citect Explorer с открытой папкой Alarms*

Двойной щелчок по любой из представленных в поле *Contents* иконок вызывает появление на экране соответствующего диалога конфигурирования аларма. На рисунках 3.14 и 3.15 приведены диалоги для конфигурирования дискретного и аналогового алармов.

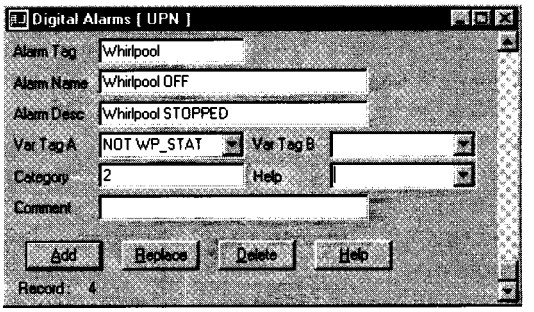


Рис. 3.14. *Диалог для конфигурирования дискретного аларма*

Хотелось бы обратить внимание читателя на поле *Var Tag A*. Имени переменной WP_STAT предшествует логический оператор NOT – значит, дискретный аларм сработает на FALSE/OFF.

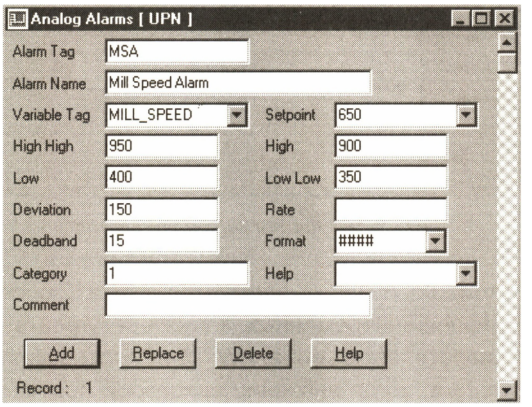


Рис. 3.15. *Диалог для конфигурирования аналогового аларма*

Каждый тип аларма обладает специфическими параметрами 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 одного и того же типа достаточно один раз войти в соответствующий диалог и произвести конфигурирование всех алармов.

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. 3.16).

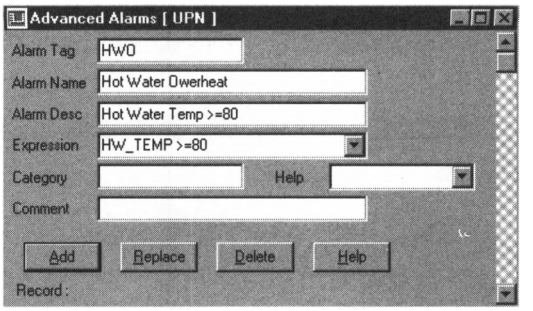


Рис. 3.16. *Диалог Advanced Alarms*

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 классификации алармов по самым различным признакам: участкам производства, 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, он 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 имеет 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. 3.17.

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).

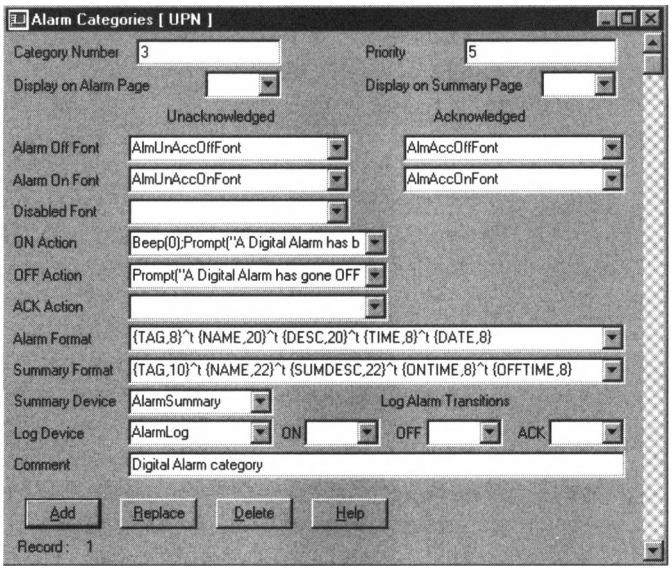


Рис. 3.17. *Диалог Alarm Categories*

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. Так же, 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. 3.18).

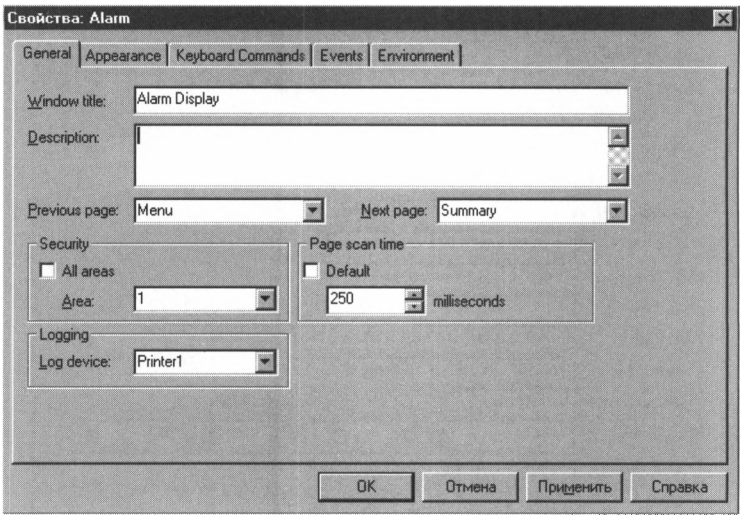


Рис. 3.18. *Диалог Properties (Свойства страницы)*

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 отображаться на страницах алармов.

Пример страницы текущих алармов *Alarm Display* приведен на рис. 3.19.

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.

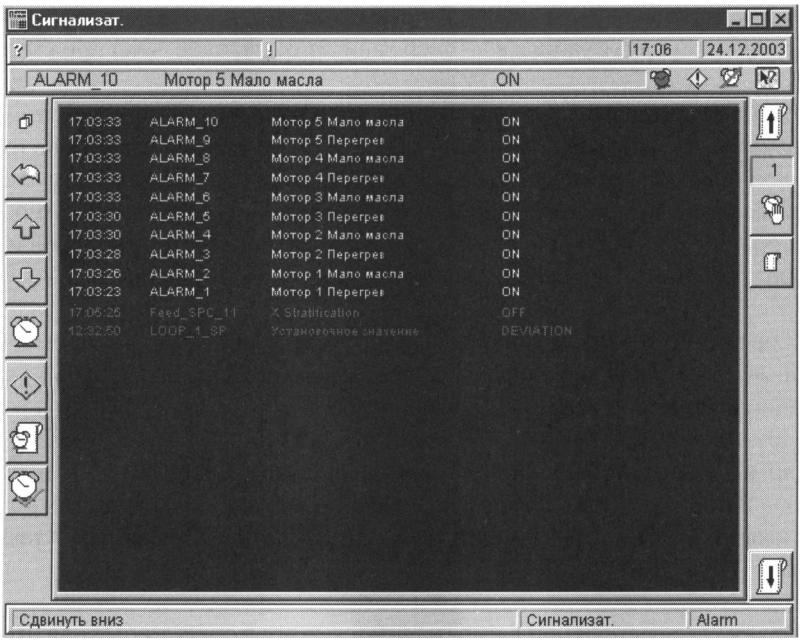


Рис. 3.19. *Страница текущих алармов Alarm Display*

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).

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 (кнопки перехода на другие страницы проекта, кнопки подтверждения алармов, кнопки запуска процесса регистрации алармов в файл или на принтер, линейки прокрутки и т.д.).

Следует напомнить читателю, 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. 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;

- 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 технологических процессов 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 просмотра прошлых значений. Отображаемые трендом данные в таком режиме – неподвижны, а их объем ограничен начальным и конечным временем выборки.

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. Например, 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).

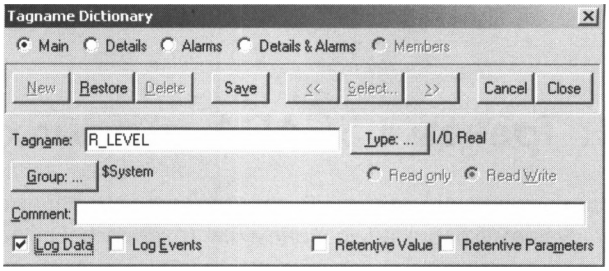


Рис. 4.1. *Диалог Tagname Dictionary с отмеченной опцией Log Data*

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. 4.2).

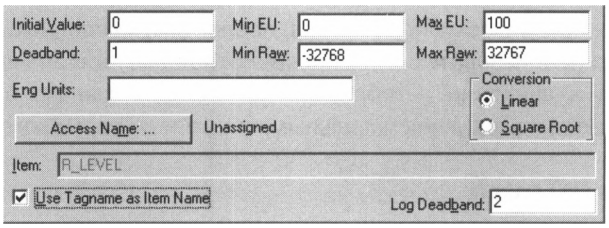


Рис. 4.2. *Диалог детального описания вещественной переменной*

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. 4.3), который вызывается на экран командой *Special/Configure/Historical Logging*. В этот диалог можно также войти из окна *Application Explorer*.

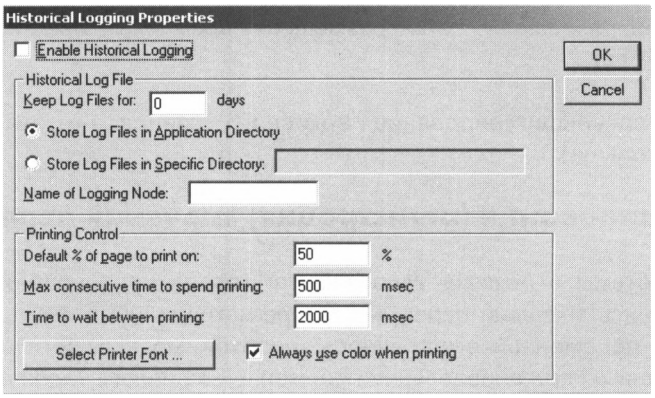


Рис. 4.3. *Диалог Historical Logging Properties*

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 – 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 значения переменных или выражений, содержащих одну или несколько переменных, в момент их изменения.

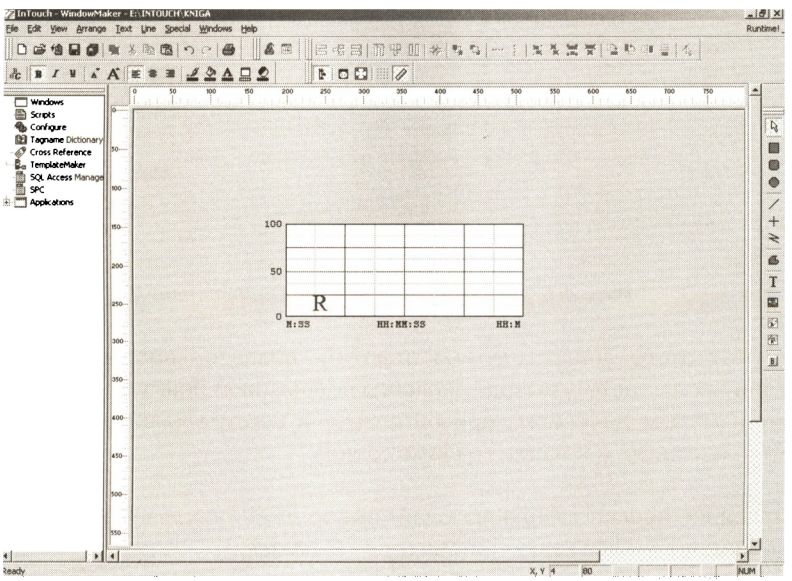


Рис. 4.4. *Объект «Тренд реального времени»*

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. 4.4).

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 (рис. 4.5) *Real Time Trend Configuration* (Конфигурирование тренда реального времени).

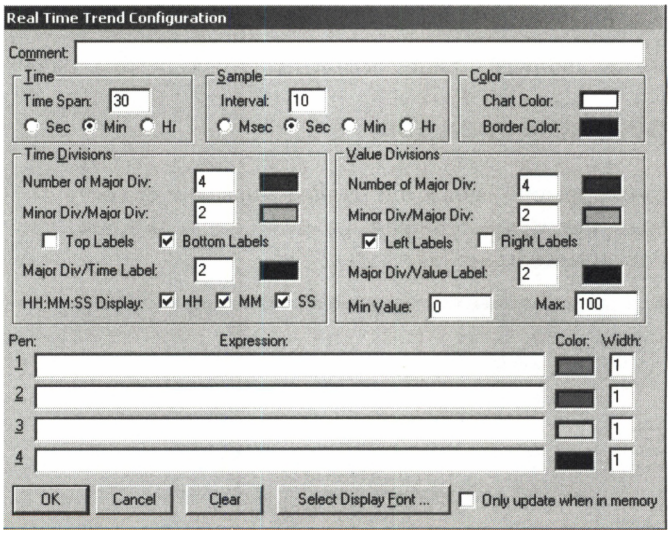


Рис. 4.5. *Диалог Real Time Trend Configuration*

Среди настроек этого диалога можно отметить диапазон времени, охватываемый трендом (Time Span), частоту вывода значения 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 вывода 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 вывода 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 определяемом бегунком положении.

Можно создать правый и левый бегунки и производить обработку данных кривой, расположенной между бегунками. Вычисляются следующие величины: среднее, минимальное, максимальное, отношение min/max 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 вышесказанное 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. 4.6).

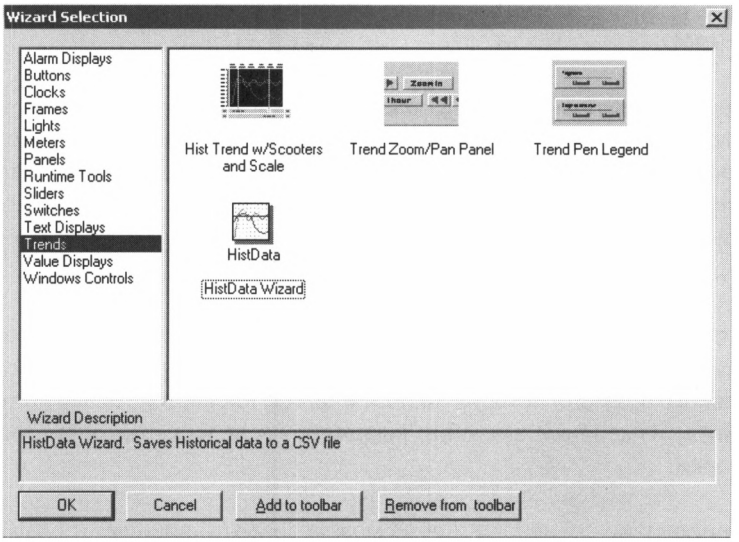


Рис. 4.6. *Диалог Wizard Selection (Выбор мастер-средств)*

После выбора из предложенного набора мастер-средств *Hist Trend with Scooters* (архивный тренд с бегунками) и щелчка по кнопке *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. 4.7). Объекты этого типа ведут себя аналогично любым другим объектам, то есть их можно перемещать, масштабировать и т. д.

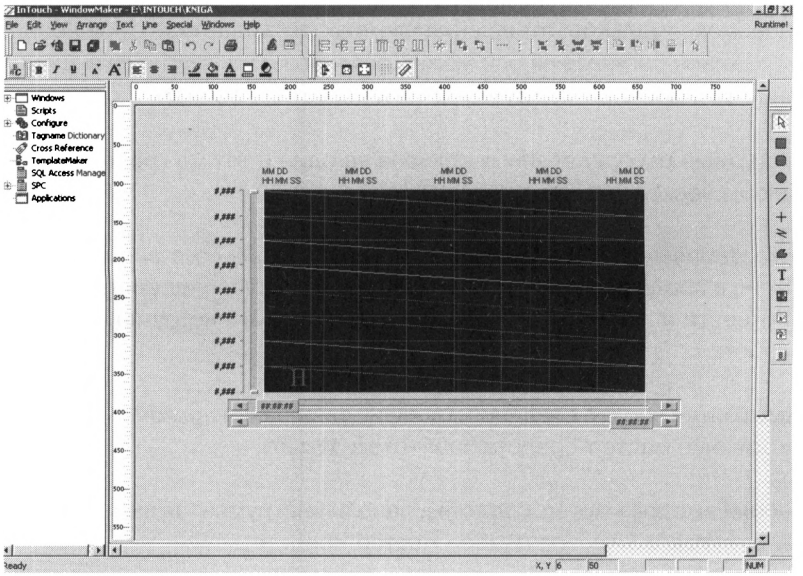


Рис. 4.7. *Объект «Архивный тренд»*

Двойной щелчок на объекте приводит к появлению на экране диалога *Historical Trend Char Window* (Конфигурирование архивного тренда, рис. 4.8).

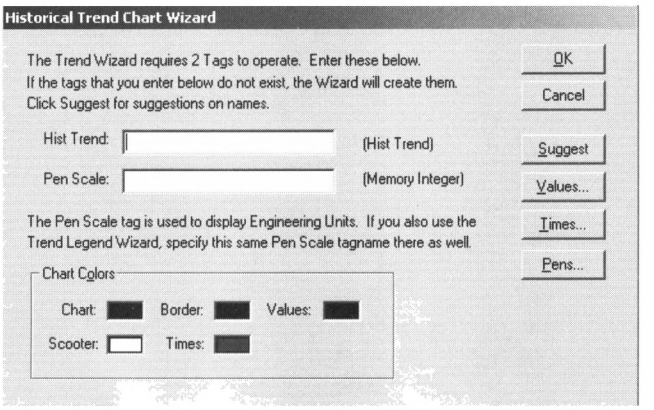


Рис. 4.8. *Диалог конфигурирования архивного тренда*

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. 4.6) соответственно. Для того чтобы эти компоненты работали совместно, они должны иметь одинаковые имена (*Hist Trend*).

4.1.3. Изменение параметров архивных трендов в режиме исполнения

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. 4.9). 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.

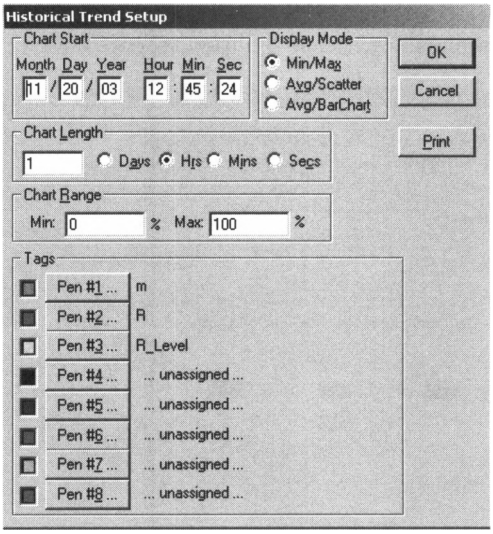


Рис. 4.9. *Диалог изменения параметров архивного тренда*

Архивный тренд может выводиться в одном из трех возможных режимов:

- **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. 4.10, 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.

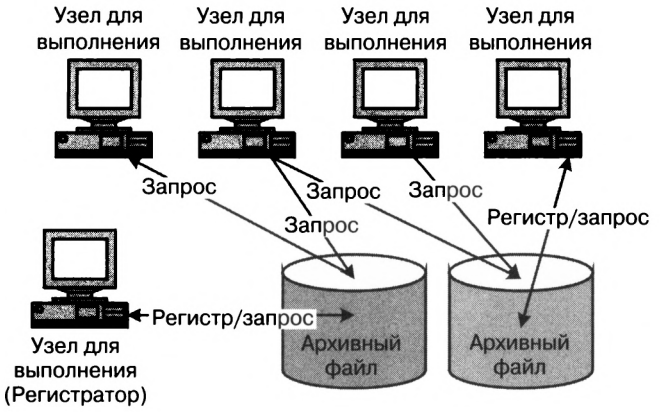


Рис. 4.10. *Распределенная система архивов*

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.

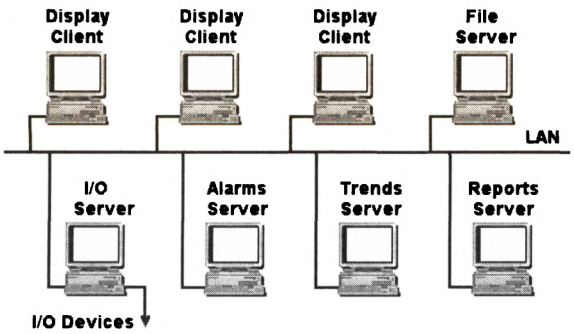


Рис. 4.11. *Вариант сетевой архитектуры системы Citect*

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.

Citect предоставляет возможность вывести на тренд любую переменную или значение выражения на языке Cicode. На одном экране допускается размещать любое количество трендов, а в каждом окне тренда можно графически отобразить до восьми переменных. 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. 3.13). 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. 4.12).

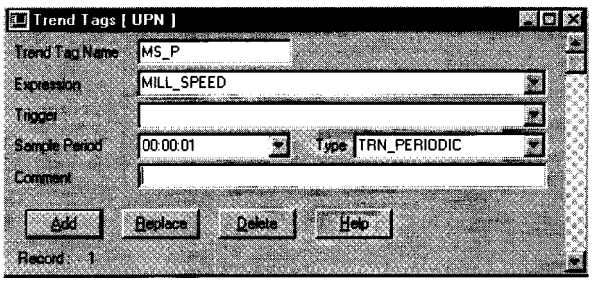


Рис. 4.12. *Диалог конфигурирования трендов*

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).

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. 4.13). 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.

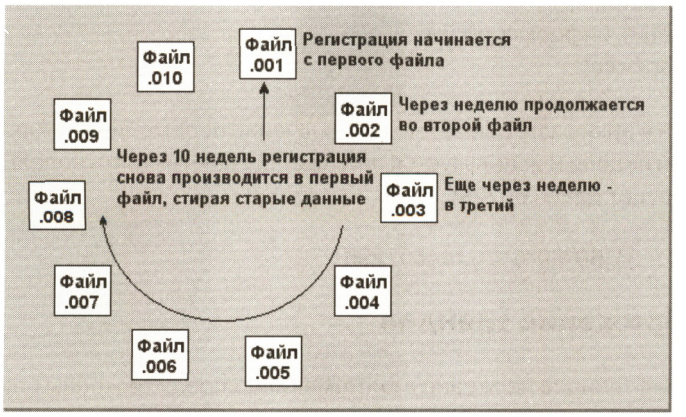


Рис. 4.13. *Круговая система записи данных в файлы*

По умолчанию имя файла будет содержать 8 символов имени переменной тренда.

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. 4.14).

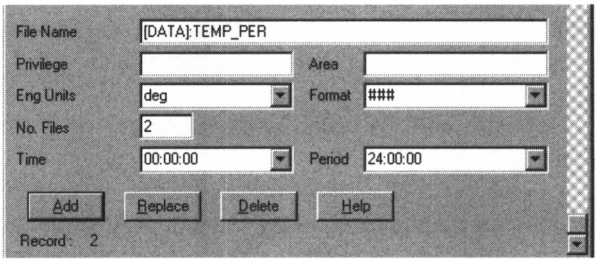


Рис. 4.14. *Диалог настройки параметров регистрации данных в файл*

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 20th April 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 (с);
t – период выборки (с).

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).

- **тренд 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 новом шаблоне, 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. 4.15).

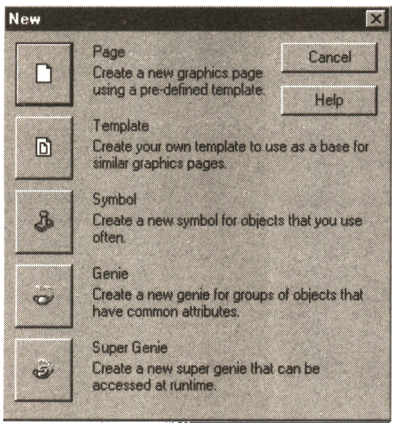


Рис. 4.15. *Меню New для создания новой страницы, шаблона, символа, джинна и суперджинна*

Создание нового шаблона – интересная, творческая работа. Но читателю не менее важно 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. 4.16.

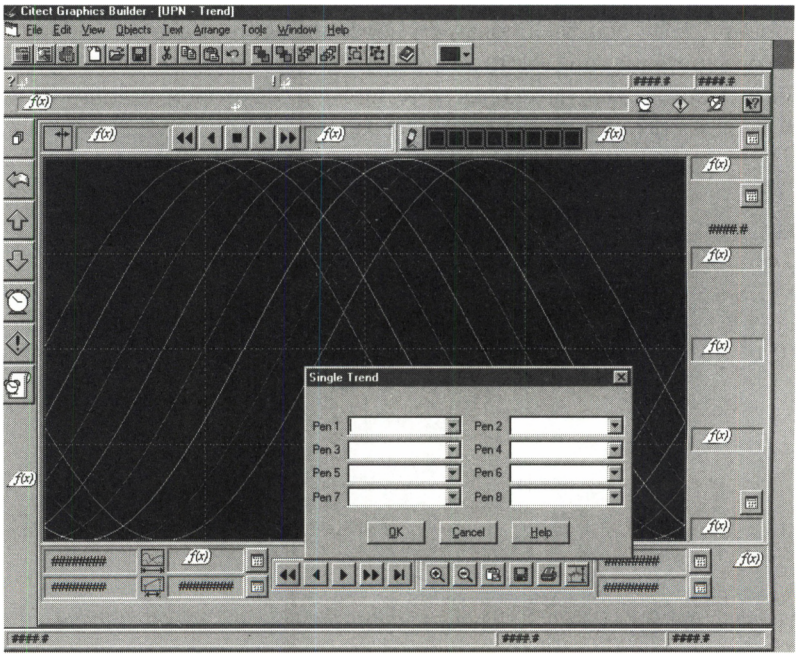


Рис. 4.16. *Шаблон одиночного тренда с окном настройки перьев*

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. 4.17).

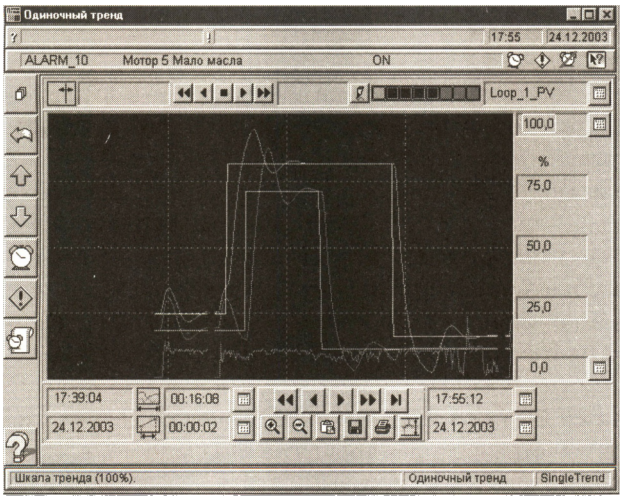


Рис. 4.17. *Страница одиночного тренда в режиме Runtime*

4.3. Отличия подсистем архивирования и отображения 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 указывается количество файлов в буфере и продолжительность регистрации в файл, в InTouch создается один архивный файл и указывается продолжительность регистрации в него.

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 (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 для доступа к ним предоставляются утилиты.

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 Citect считает подобные шаблоны типичными для использования в проектах (свой опыт компания выразила в шаблонах) и предлагает их своим пользователям. 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 собы-

тий), 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 возникают 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. 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 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 приведен диалог *Application Scripts* (Скрипты уровня приложения).

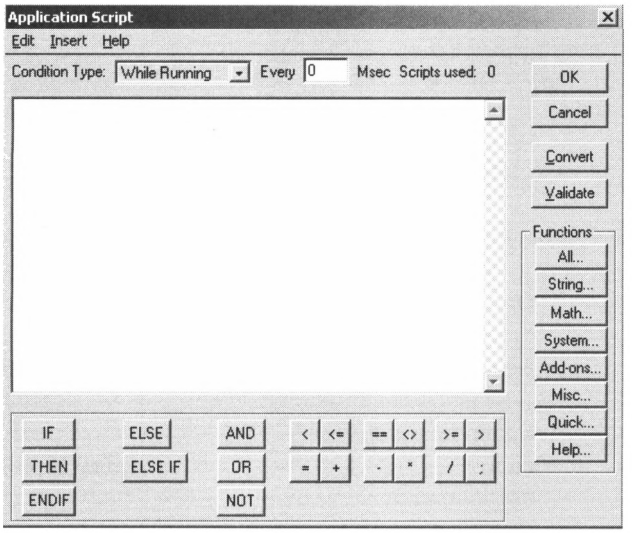


Рис. 5.1. *Редактор скриптов Application Scripts (Скрипты уровня приложения)*

Редактор скриптов InTouch поддерживает два типа скриптов: **простые** и **сложные**. Простые скрипты содержат 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).

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. 5.2).

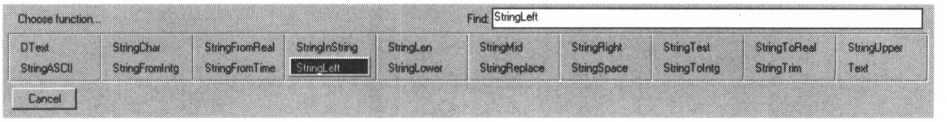


Рис. 5.2. *Список строковых функций*

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. 5.3).

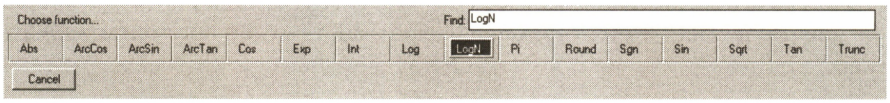


Рис. 5.3. *Список математических функций*

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. 5.4.

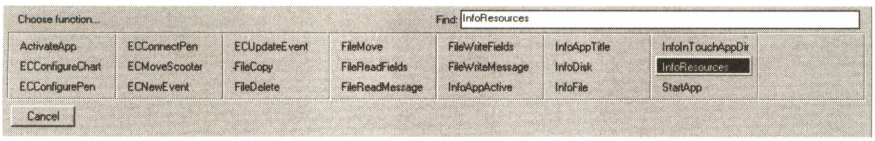


Рис. 5.4. *Список системных функций*

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 5.2

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 аларма-ми распределенных систем, трендами, печатью и др. (рис. 5.5).

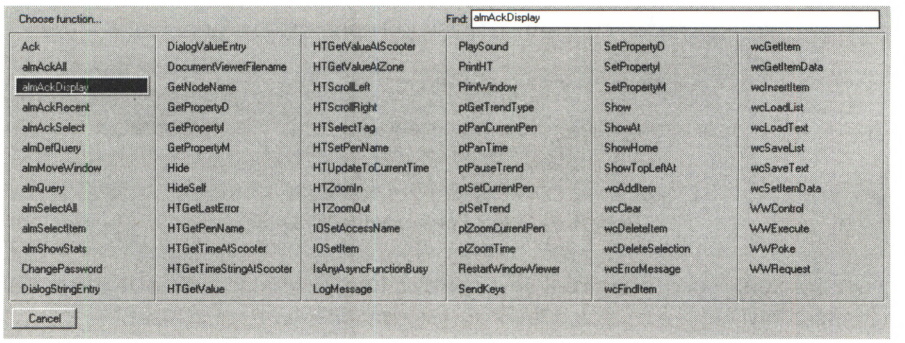


Рис. 5.5. *Список функций группы Miscellaneous*

В этой широкой (с точки зрения назначения функций) группе можно выделить несколько специализированных подгрупп. Функции, название которых начинается с alm, используются только в распределенных системах алармов. Некоторые из них приведены в табл. 5.3.

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

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. 5.4.

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 нового времени начала и охватываемого интервала времени

Таблица 5.4

Встроенные функции для работы с архивными трендами также могут иметь несколько аргументов (до четырех). Функции, приведенные в табл. 5.4, 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. 5.5), 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 5.5

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* (Печать окна) для печати окна. Ее синтаксис выглядит следующим образом:

PrintWindow(«Window», Left, Top, Width, Height, Options);

где:
Window – имя окна;
Left – количество дюймов от левого края;
Top – количество 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 Functions

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. 5.6.

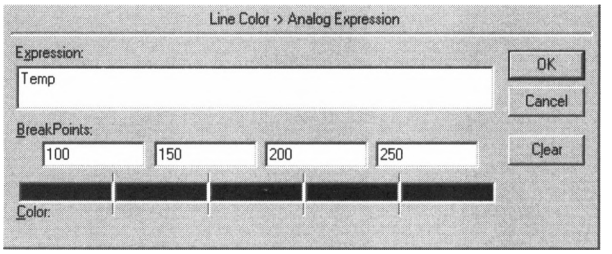


Рис. 5.6. *Диалог Fill Color (Цвет заполнения)*

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 командой *Special/Scripts* с последующим нажатием на строке *Quick Functions* (рис. 5.7).

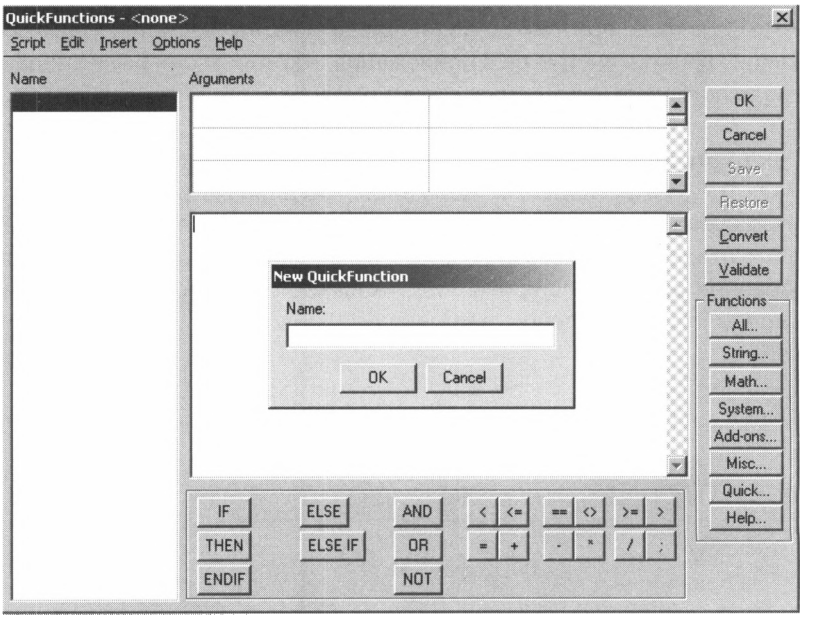


Рис. 5.7. *Диалог редактора Quick Functions*

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. 5.7). После щелчка по кнопке *Ok* новое имя будет включено в список *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 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 использования Touch Commands. В графических интерфейсах часто применяют кнопки для запуска и остановки насосов, для вкл./выкл.

электродвигателей, для перехода на другие графические страницы. Характерным примером использования Touch Commands является вызов выпадающего окна для ввода информации (суперджинн). 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 (Keyboard Commands)

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);
- 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. 5.8).

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.

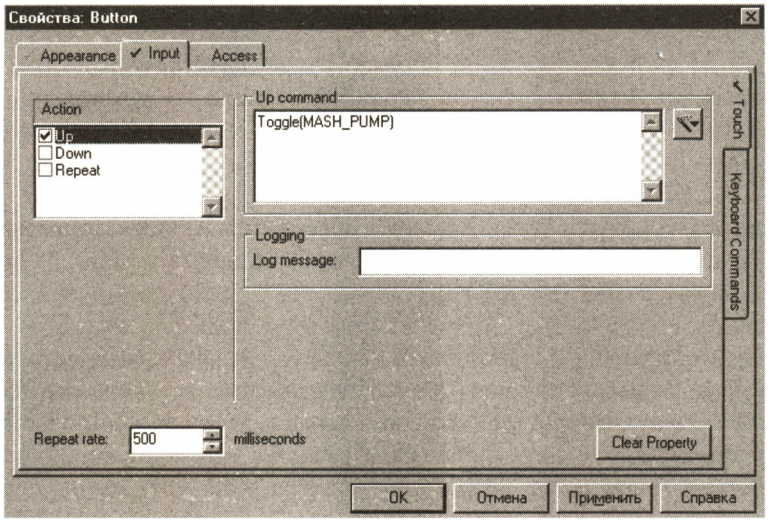


Рис. 5.8. *Диалог «Свойства объекта» на закладке Input/Touch*

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 требуют нескольких аргументов. Список аргументов должен находиться в скобках, один аргумент отделяется от другого 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.

Примером такой функции является AlarmAck(1, 0);. С ее помощью 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.

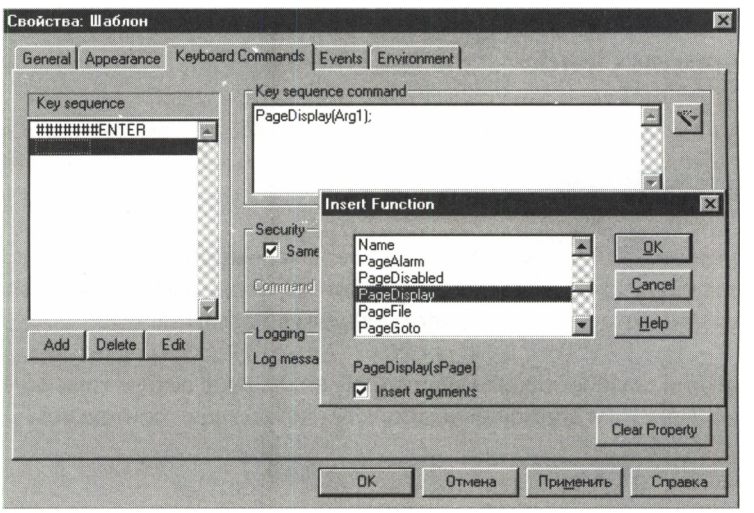


Рис. 5.9. *Диалог «Свойства страницы» на закладке Keyboard Commands*

В качестве аргумента в функции можно использовать последовательность клавиш, вводимую оператором в режиме исполнения. Например, для облегчения выбора оператором страниц проекта (рис. 5.9) можно определить последовательность клавиш ####### Enter и команду PageDisplay(Arg1);.

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 шесть 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 насчитываются несколько десятков групп встроенных функций. Основные из них приведены в табл. 5.6.

Группа функций	Описание
ActiveX	Вызывают и взаимодействуют с ActiveX-объектами
Alarm	Управляют 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 ввода/вывода
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 5.6

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. 5.7.

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 5.7

Группа Page насчитывает более 20 функций, среди которых часто применяются следующие функции (табл. 5.8).

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 5.8

Аргументы приведенных в табл. 5.8 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 решаются 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*.

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. 5.10). Одновременно может быть открыто несколько таких окон с программами, принадлежащими различным проектам.

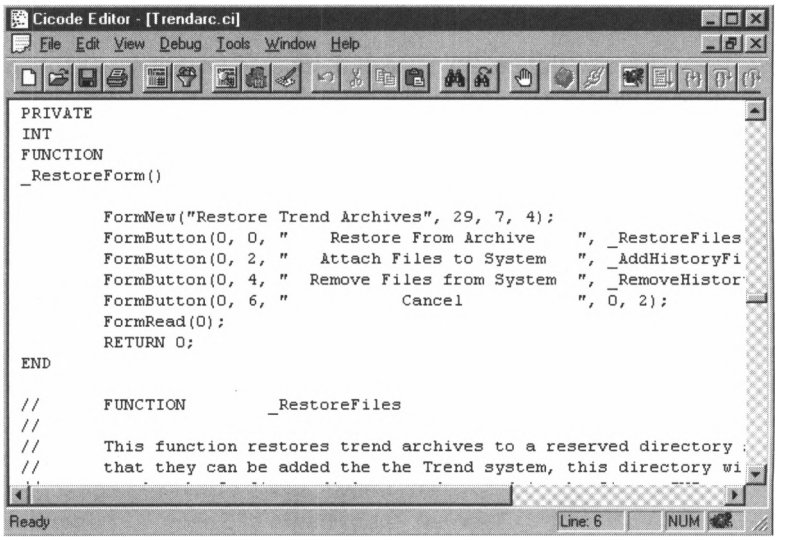


Рис. 5.10. *Окно Редактора Cicode (Cicode Editor)*

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: красный – в режиме редактирования программы, зеленый – в режиме отладки.

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. База данных

В самом общем смысле база данных (БД) – это система хранения информации, обращение к которой осуществляется через средство управления базой данных (СУБД). Практически база данных представляет собой набор данных, рассортированных по уникальным идентификаторам и организованных 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 справляются 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. Поэтому 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.

Решение перечисленных выше проблем оказалось возможным с появлением иерархических БД. В таких базах элементы данных строго упорядочены, причем так, что данные одного уровня подчиняются (являются подмножеством) данным другого, более высокого уровня. В такой модели связи данные 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** – **O**nline **T**ransaction **P**rocessing) и системы поддержки принятия решений (**DSS** – **D**ecision **S**upport **S**ystem).

OLTP-системы используются для создания приложений, поддерживающих ежедневную активность организации. Обычно эти приложения являются критическими для деятельности и поэтому требуют быстроты отклика и жесткого контроля над безопасностью и целостностью данных..

DSS-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** – **O**nline **A**nalytical **P**rocessing).

Критерии оценки БД

Базы данных будут продолжать развиваться, а объемы информации в компьютерах – расти. Усложнение производственных процессов, «интеллектуализация» контрольно-измерительных приборов, требования конечного пользователя относительно повышения объемов и качества информации делают это предположение особенно справедливым для промышленных условий.

Однако наиболее важные критерии оценки БД останутся теми же самыми, а именно:

- повышает ли БД возможности конечных пользователей путем предоставления доступа к нужной информации в нужном месте и в нужное время?
- обеспечивает ли БД требуемый уровень открытости и гибкости запросов?
- легко ли сопровождать и использовать БД, надежна ли она?
- широко ли распространена БД и хорошо ли поддерживается ее технология большим числом независимых производителей программного обеспечения?
- легко ли интегрировать БД с широким спектром иного программного обеспечения?
- широк ли спектр возможных применений БД?
- доступны ли по цене большинству пользователей аппаратные платформы, поддерживаемые БД?
- приемлема ли сама БД по цене для большинства пользователей?

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** (**S**tructured **Q**uery **L**anguage) – язык структурированных запросов. Он представляет собой средство организации, управления и поиска информации в РБД. Широкое признание SQL приобрел благодаря таким 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.

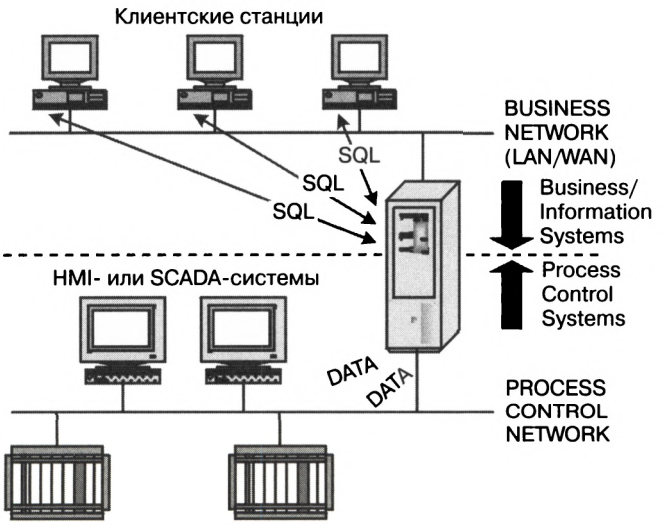


Рис. 6.1. *Клиент-серверная организация*

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, ремонтникам, 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, обладающее 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 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 FactorySuite2000. Несмотря на то, что IndustrialSQL Server поставляется компанией Wonderware как самостоятельный продукт, он одновременно является одним из главных компонентов пакета FactorySuite2000, 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 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 используются сервером Microsoft SQL Server.

Выбор Microsoft SQL Server в качестве основы для IndustrialSQL Server объясняется несколькими причинами. Во-первых, в мире существуют 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.

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.

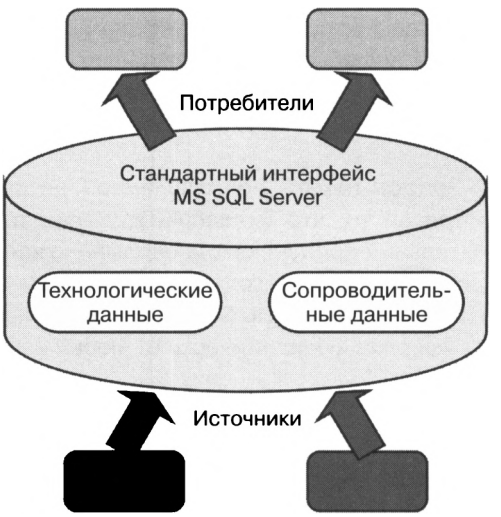


Рис. 6.2. *IndustrialSQL Server на основе MS SQL Server*

На рис. 6.2 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.

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 мощному и гибкому процессору запросов пользователи могут 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. IndustrialSQL Server, являясь сервером БД в составе пакета FactorySuite 2000, 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 контролирующих 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 понимания состояния и хода 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 средствами фильтрации, объединения и обработки данных.

Конфигурационные параметры, как и вся предыстория модификаций, хранятся в «чисто» Microsoft SQL-таблицах, доступных через SQL. В процессе функционирования предприятия могут добавляться новые и удаляться существующие параметры, меняться описания и диапазоны измерений.

Сохранение предыстории модификаций гарантирует соответствие конфигурационных параметров возвращаемым сервером архивным данным.

Сервер реального времени

В язык запросов IndustrialSQL Server включены средства работы с временными характеристиками данных. Входящие в состав Wonderware FactorySuite серверы ввода/вывода используют новый протокол SuiteLink. В этом протоколе впервые была введена концепция отметок времени и качества информации, выставляемых серверами ввода/вывода. Кроме того, благодаря протоколу SuiteLink удалось еще больше 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.

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 упрощаются 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 незапрошенных 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

Одним из достоинств IndustrialSQL Server является 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. 6.3):

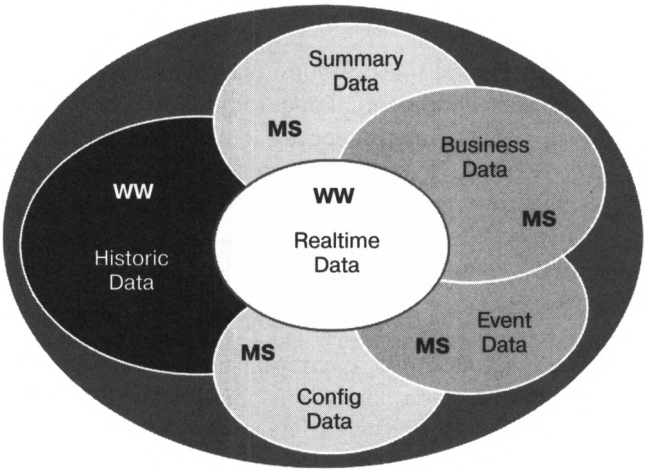


Рис. 6.3. *Типы данных, регистрируемых IndustrialSQL Server*

- реального времени;
- архивные;
- конфигурационные;
- сводные;
- сопутствующие учрежденческие.

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 копиро-

вания 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

База данных реального времени IndustrialSQL Server является важной составляющей пакета FactorySuite 2000 и легко интегрируется 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.

6.1.3. Область 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 многие вопросы: 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? и пр.

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 и оперативные данные.

Руководители 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 следующие вопросы: каков объем дневного выпуска продукции? каковы причины и длительность простоев оборудования в этом месяце? соответствует ли выпуск продукции плановым показателям? и пр.

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 разные вопросы: 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? и пр.

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 Citect

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 доступны самые последние данные технологического процесса, что позволяет им принимать решения, полностью владея информацией о процессе производства.

Большинство SCADA-систем могут 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 типа Microsoft ActiveX используются для упрощения интеграции Plant2SQL с такими пакетами, 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. 6.4)

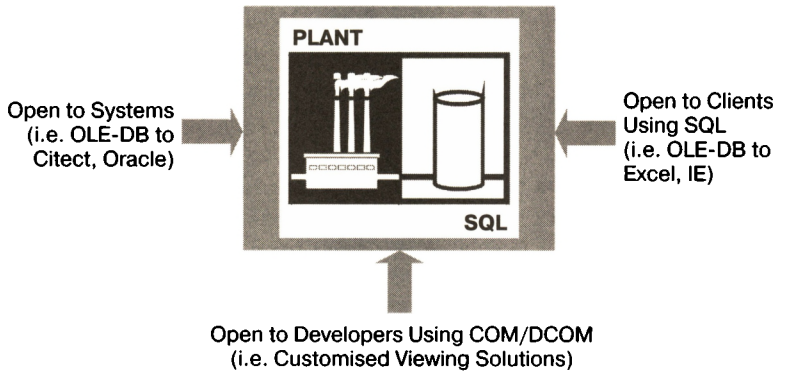


Рис. 6.4. *Протоколы доступа к Plant2SQL*

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 совместим с ним на 100% (рис. 6.5).

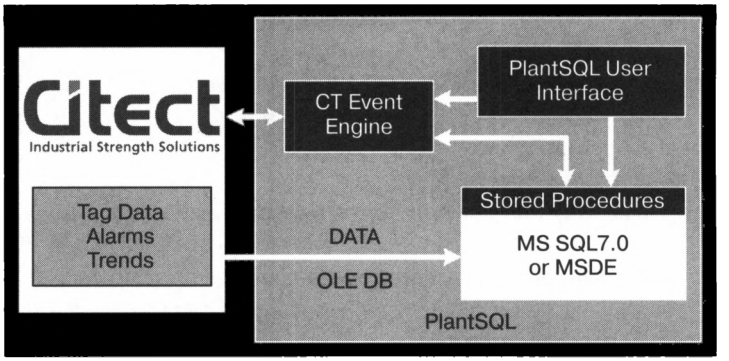


Рис. 6.5. *MS SQL Server – основа Plant2SQL*

По умолчанию все трендовые и алармовые данные автоматически доступны клиентскому приложению. Пользователи могут выбрать список переменных, которые регистрируются в базе данных Microsoft SQL, и затем просматривать их значения в реальном времени или в ретроспективе.

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 так, чтобы включать любую группу переменных с возможной записью в эти переменные.

Архитектура. Plant2SQL имеет различные опции расширения. В малых простых приложениях возможен запуск сервера и клиента Plant2SQL 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-клиенты позволяют нетехническим 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, можно 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. 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) может легко отследить количество часов работы оборудования, чтобы заранее запланировать и вовремя осуществить его диагностику.

Менеджеры по производству могут легко интегрировать бизнес-информацию с технологической и быстро просчитывать стоимость инвестиций и материальных издержек.

6.3. Базы данных реального времени. Их отличия

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 БДРВ;

- 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 по-разному:

- 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 с помощью 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 данных по DDE-, SuiteLink-протоколам.

ГЛАВА 7. Internet/Intranet-решения и SCADA-системы. Стратегия клиентских приложений

Тема обеспечения доступа к данным технологического процесса с любого компьютера предприятия, с любой подсистемы стала актуальной. Не располагая информацией, в частности информацией реального времени, невозможно эффективно управлять предприятием. SCADA-приложения, по определению, являются потребителями технологических данных, но, с другой стороны, они должны быть и их источником. Информация со SCADA-приложений потребляется многочисленными клиентами (специалистами и руководителями).

Вопросам организации доступа к информации и посвящается настоящая глава. При изложении материала этой главы будет использован термин «клиентское приложение». В самом общем смысле под этим термином следует понимать программное обеспечение сетевого компьютера, необходимое для получения удаленного доступа к производственной информации в соответствии с имеющимися правами.

Рассматривая различные типы клиентских приложений, нельзя оставить без внимания и протоколы, используемые для передачи как исторических данных, так и данных реального времени. В конце главы рассмотрен специальный инструментарий для создания Internet/Intranet-клиентов.

Если необходимость организации клиентских узлов для удаленного доступа к технологической информации не вызывает сомнения, то ответ на вопрос «как их организовать?» пока остается открытым.

Каждый клиентский узел реализует вполне определенные функции. И поэтому клиентское приложение должно обеспечить соответствующий данному клиентскому узлу набор пользовательских услуг. К их числу можно отнести:

- объем предоставляемой информации;
- форму представления информации;
- реализуемые функции (только информационные или с возможностью выдачи управляющих воздействий);
- протяженность и надежность канала связи «источник-потребитель»;
- простоту.

Клиентские приложения различного типа могут предоставлять информацию в любом объеме и приемлемом для пользователя виде.

Клиент-серверная организация SCADA-систем предполагает применение клиентских приложений двух типов: с возможностью передачи управляющих воздействий с клиентского приложения и чисто мониторинговые приложения. Пользователю достаточно лишь определить необходимый набор услуг.

Но за услуги, как известно, надо платить. Поэтому весьма существенным критерием при организации клиентского узла является его стоимость (аппаратное и программное обеспечение).

В настоящее время существует несколько решений поставленной задачи, базирующихся на применении различных технологий. Но и стоимость предлагаемых решений тоже различна. Отсюда и появились такие понятия, как «бедные/богатые» и «тонкие/толстые» клиенты. В клиент-серверной архитектуре большая часть исполняемого кода «тонкого» клиента реализуется на сервере. Для «толстого» клиента практически весь код реализуется на компьютере клиента; с серверной части поставляются, в основном, обновленные данные. В отличие от «толстого», «богатый» клиент в большей степени ссылается на применяемые при создании пользовательского интерфейса технологии, чем на то, какое количество кода выполняется на стороне клиента. «Богатые» клиенты похожи на обычные приложения Win32, но они представляют собой клиентскую часть трехуровневого приложения.

Самыми простыми и распространенными клиентскими приложениями являются клиенты в локальной сети (рис. 7.1). Такие клиентские компоненты в SCADA-системах традиционно объединяются с серверными приложениями протоколами локальных сетей (TCP/IP, NetBEUI).

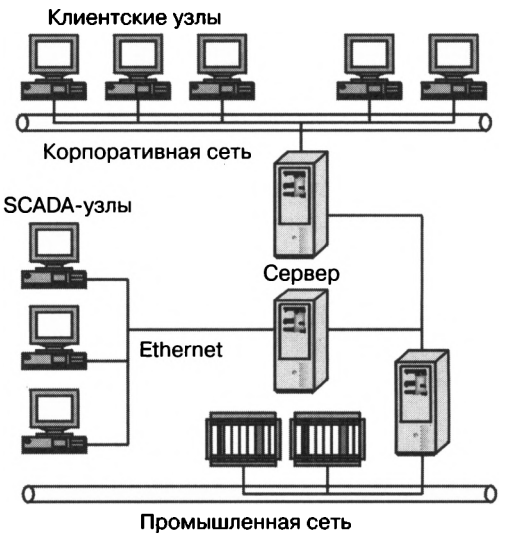


Рис. 7.1. *Традиционное решение*

Но заявленная простота может достаточно дорого обойтись. Не все клиенты сети нуждаются в информации реального времени. Более того, для многих клиентов

требуются только итоговые технико-экономические показатели, т. е. информация, не критичная ко времени. А для реализации предложенной на рис. 7.1 структуры может потребоваться установка на компьютеры, поддерживающие клиентские приложения, различных компонентов SCADA-систем, а следовательно, и дополнительные лицензии.

Бурно развивающиеся в последние годы Internet/Intranet-технологии не могли не привлечь внимание разработчиков SCADA-систем, баз данных реального времени. В результате появились новые типы 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** (**D**istributed **I**nternet **A**rchitecture). Поэтому предлагается начать с краткого изложения особенностей этой структуры.

7.1. Структура Windows DNA

Структура Windows DNA – это в первую очередь реализация трехуровневой модели приложения, включающей (рис. 7.2):

- 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 так называемые «склеивающие» технологии. В программном обеспечении Microsoft роль «склеивающих» технологий играют COM и COM+. **COM** (**C**omponent **O**bject **M**odel – архитектура компонентных объектов) – это объектно-ориентированная технология.

Приложение с компонентной организацией 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: «бедный» (thin) и «богатый» (rich). «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. При этом требуется дорогостоящий сервер, широкая полоса пропускания между клиентом и сервером. Чаще всего понятие *бедный клиент* обозначает приложение, работающее на Web-сервере и передающее пользовательский интерфейс с помощью HTML-страниц на Web-браузер.

Позднее возникла идея обогащения Web-приложений различными компонентами, которые могут использоваться браузером: управляющими элементами ActiveX, апплетами Java и т.д. «Бедные» клиенты различной оснащенности предлагаются и компаниями-поставщиками SCADA-систем.

Уровень бизнес-логики. Три сервиса свойственны этому уровню: сервисы компонентов (COM), **M**icro**s**oft **M**essage **Q**ueue (**MSMQ**) и **I**nternet **I**nformation **S**erver (**IIS**). Internet Information Server – 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 – атомарность. Именно концепция транзакции обеспечивает выполнение ряда операций получения данных из разных СУБД и позволяет рассматривать их как единую операцию (рис. 7.3).

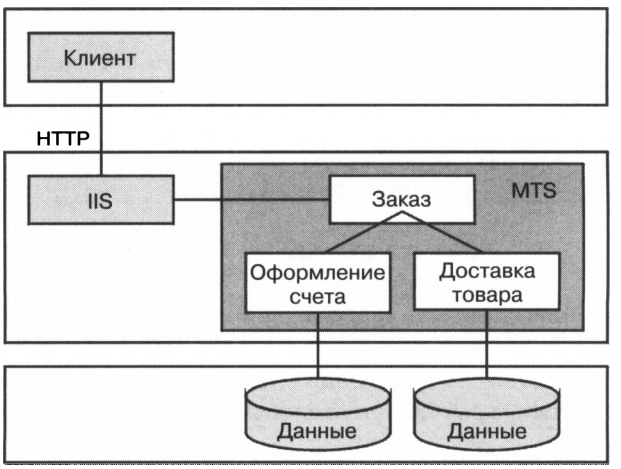


Рис. 7.3. *Трехуровневое приложение*

Microsoft Message Queue – асинхронная однонаправленная связь, ориентированная на сообщения. Как **DCOM** (**D**istributed **COM**), 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 (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 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** (**R**emote **D**esktop **P**rotocol) и Citrix **ICA** (**I**ndependent **C**omputing **A**rchitecture), что допускает реализацию клиентов в виде супертонких бездисковых рабочих станций на платформах Linux/CE от Windows 3.11/95/98 до рабочих станций Windows NT/2000.

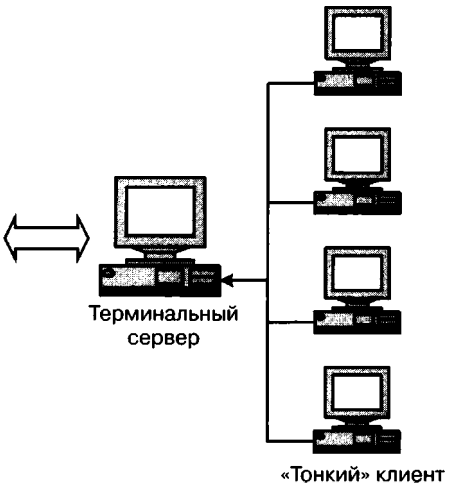


Рис. 7.4. *Архитектура «терминал-сервер»*

Используя новые архитектурные возможности, компании-разработчики SCADA-систем могут 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 как клиентские приложения. 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. 7.4).

Терминальные пользователи имеют доступ к данным, графическим мнемосхемам с возможностью обмена информацией в реальном времени без необходимости установки InTouch на локальном клиентском компьютере.

Применение терминал/серверной модели позволяет создавать более экономичные решения за счет того, что:

- приложения устанавливаются и поддерживаются инженерами только на сервере;
- терминальные клиенты могут быть реализованы на различных платформах.

Следует заметить, что на клиентских узлах можно просматривать как одно и то же приложение, так и разные.

7.3. Стратегия клиентских приложений от Wonderware

7.3.1. «Бедные» и «богатые» Internet/Intranet-клиенты

В Internet/Intranet-решениях по обмену данными, кроме технологического сервера как поставщика данных и клиента как получателя информации, задействован Web-сервер (рис. 7.5).

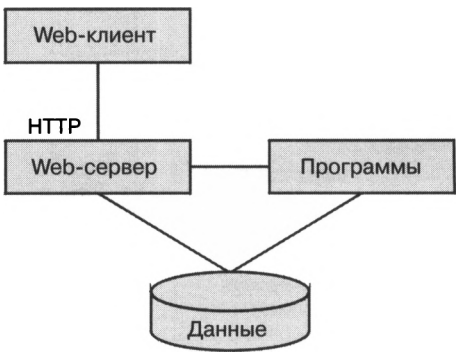


Рис. 7.5. *Клиенты и серверы Web*

Информация на сервере хранится в виде страниц, на которых, кроме текста, могут находиться разные объекты: графические изображения, аудио- и видеоролики, формы

для ввода данных, интерактивные приложения и т.д. Страницы сервера **WWW** (**W**orld **W**ide **W**eb) могут содержать не только статическую, но и динамическую информацию.

Страница может содержать формы для выполнения запросов к базе данных. Результат такого запроса будет динамически сформирован в виде страницы, которая появится на экране пользователя.

Сервер WWW может решать любую задачу, принимая любые данные от удаленного пользователя, обрабатывая их и передавая обратно.

Для обработки на сервере WWW запросов, поступающих от клиентских приложений SCADA и требующих получения данных из БДРВ или других источников информации РВ, разрабатывается специальное серверное расширение, которое, с одной стороны, получает и обрабатывает динамические запросы от Web-клиентов, а с другой – обеспечивает взаимодействие с Microsoft Internet-серверами.

Взаимодействие между Web-сервером и клиентами осуществляется на основе протокола **HTTP** (**H**yper**T**ext **T**ransfer **P**rotocol – протокол передачи гипертекста). Так, компанией Wonderware предлагается **F**actory**S**uite (**FS**) Web-сервер, который обеспечивает динамическими данными Web-клиента, реализованного в виде SCADA-приложения InTouch (приложение 7.1).

На рисунке 7.6 показаны возможности разработки Internet-приложений и их запуск в реальном времени на примере SCADA-системы InTouch.

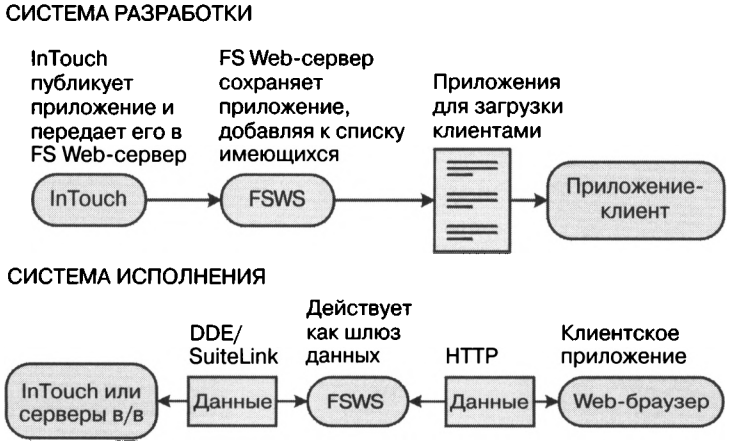


Рис. 7.6. *Web-сервер для обмена данными между приложениями InTouch*

Следует отметить, что процедура публикации (publishing) SCADA-приложений является дружественной и не требует специальной подготовки (приложение 7.2).

Для просмотра приложения Web-клиентом могут использоваться навигатор Microsoft Internet Explorer или исполняющая система InTouch.

Интернет-приложение позволяет собирать данные с многих FS Web-серверов (рис. 7.7).

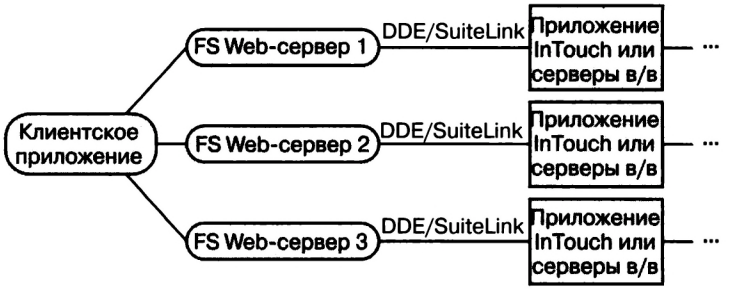


Рис. 7.7. *Получение данных от нескольких Web-серверов*

В таких случаях каждому Web-серверу присваивается специальное имя или IP-адрес. Чтобы подписаться на приложение, необходимо загрузить его из текущего FS Web-сервера и выделить в локальную директорию на клиентской машине.

Публикация InTouch-приложения возможна в двух режимах: с исходными файлами, так что приложение в дальнейшем может модифицироваться в среде разработки, и только в режиме исполнения.

Приложение 7.1

Серверное расширение FactorySuite Web Server

Именно эта компонента в FactorySuite версии 7.1 является новым решением, соответствующим современным Internet/Intranet-технологиям. FS Web Server устанавливается на Microsoft Internet Information Server (IIS) или Personal Web Server(PWS).

Для просмотра и управления приложением через Internet рекомендуется использовать Microsoft Internet Explorer версии 4.0 SP1 и более поздних версий. FS Web Server позволяет:

- запускать InTouch-приложения через Internet/Intranet;
- запускать существующие или создавать новые InTouch-приложения Internet/Intranet;
- передавать статические и динамические данные. HTTP (HyperText Transfer Protocol – протокол передачи гипертекста) используется для передачи данных между сервером WWW и клиентами;
- MS Explorer или исполняющая система InTouch могут использоваться для просмотра приложения.

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. Рассмотрение клиентов такого типа предлагается начать с клиентов к базам данных (БД).

Приложение 7.2

Опубликованные приложения могут редактироваться на компьютерах-клиентах. Для запуска приложения InTouch на клиентском узле должна быть установлена исполняющая система InTouch. Причем Web-клиентам передаются только данные, и данные следующих типов:

- удаленные InTouch-переменные;
- распределенные алармы;
- архивные данные InTouch;
- данные с серверов ввода-вывода;
- данные из IndustrialSQL Server.

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

Поскольку БДРВ поддерживают язык SQL-запросов, то для организации доступа к технологической информации возможен стандартный подход (как к обычным реляционным БД). Традиционный подход позволяет получать данные из БД и БДРВ, используя уже ставшие стандартными SQL-объекты, доступные, практически, из любого браузера. Этот подход требует программистского опыта разработки Web-сайтов и использования специальных SQL-объектов. И часто такой подход применяется для разработки «бедных» клиентов.

Ниже рассматривается более простая, с точки зрения пользователя-разработчика сайта, процедура доступа к БДРВ на примере IndustrialSQL Server от Wonderware. IndustrialSQL Server использует трехуровневую клиент-серверную архитектуру (рис. 7.8), которая позволяет создавать Internet/Intranet-приложения. Обработка запроса на получение данных, сделанного клиентским объектом к IndustrialSQL Server, поддерживается с помощью специальных объектов Business Objects. Специальные объекты являются COM-объектами и размещаются либо на локальном компьютере, либо на Microsoft Internet Information Server.

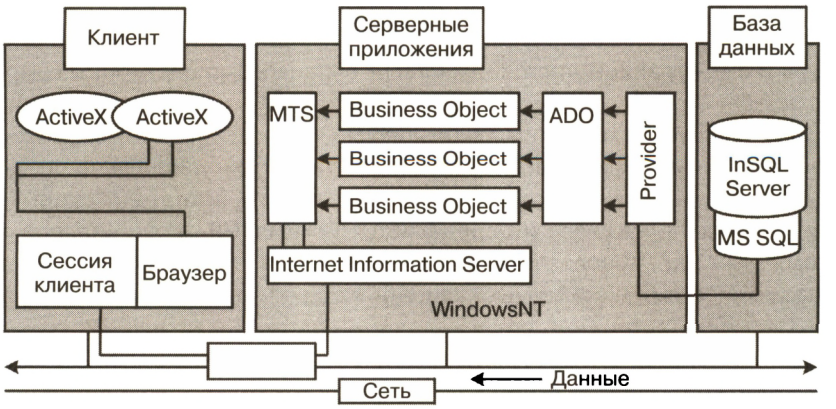


Рис. 7.8. *Трехуровневая клиент-серверная архитектура*

Клиентские приложения. Формат таблиц базы данных в БДРВ в основном предопределен. Учитывая предопределенный формат таблиц, клиентские приложения 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) и для IndustrialSQL Server компании предлагают специальные приложения, ориентированные на получение данных из БДРВ. С технологической точки зрения часть приложений реализованы как независимые, другая часть представляет ActiveX-объекты.

Приложение 7.3

Список ActiveX-объектов из FactoryOffice

- **ActiveTagBrowser** – позволяет просматривать иерархию объектов в БДРВ IndustrialSQL Server, таких, как серверы ввода-вывода, переменные, события и т.д. ActiveTagBrowser может предоставлять данные от многих IndustrialSQL-серверов. Пользовательский интерфейс для данного ActiveX-объекта аналогичен интерфейсу, предлагаемому в клиентских приложениях БДРВ.

- **ActiveDataGrid** – может быть использован для выполнения любых SQL-запросов и отображения результатов их выполнения из SQL-серверной базы данных. Данные возвращаются в табличном формате.

- **ActiveGraph** – позволяет получать исторические данные из одной или нескольких БДРВ IndustrialSQL Server и затем отображать данные в виде тренда.

- **ActiveTimeSelector** – позволяет выбирать периоды времени для выбора данных.

Как независимые приложения, встроенные в программы Microsoft Office, так и ActiveX-объекты предназначены для создания текущих и архивных трендов, параметрических графиков X-Y, а также для табличного отображения текущих и архивных данных. В приложении 7.3 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 имя 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 готовые приложения – Web-клиенты – не используются, то для создания своего Web-сайта и разработки не просто «бедного» клиента, а оснащенного ActiveX-

объектами, Java-апплетами и др., целесообразно воспользоваться существующим для этого инструментарием.

Инструментарий является разноуровневым: различают как традиционный инструментарий общего назначения, так и специализированный, ориентированный на особенности механизмов обмена в АСУТП. Специализированный инструментарий характеризуется тем, что поставляют его сейчас:

- независимые компании (Intuitive Technology). Они предлагают поддержку характерных для АСУТП протоколов (DDE, OPC, OLE DB), обеспечивая таким образом клиентские приложения данными в реальном времени;

- компании-разработчики SCADA-систем. Их инструментарий поддерживает не только ставшие стандартными протоколы обмена, но и частнофирменные протоколы, а также конвертацию приложений SCADA в HTML, XML-языки. Примером такого инструментария является SuiteVoyager от Wonderware.

Создание собственного или редактирование существующего Web-сайта. Пользователь устанавливает соединение с сервером WWW через сеть с помощью специальной программы просмотра страниц – WWW-браузера, например, навигаторов Microsoft Internet Explorer или 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 аудиофайлов, а также сценариев на **CGI** (**C**ommon **G**ateway **I**nterface) или ином языке. Для обеспечения целостности сайта используются гипертекстовые связи, hyperlink (приложение 7.4).

Приложение 7.4

Гипертекстовая связь – это способ организации связей отдельных гипертекстовых, гипермедийных или мультимедийных документов в единое целое.

Первой частью гипертекстовой связи является ссылка в виде слова, фразы, графического изображения или иной формы информации, воспроизводимой системой.

Вторая часть связи – это сгруппированная определенным образом информация или расширенное толкование исходных данных не обязательно в виде отдельного документа или файла. Гипертекстовые связи из HTML-документа могут ссылаться не только на аналогичный HTML-документ, но и на любые другие информационные файлы, которые можно найти в WWW.

Для создания сайтов предлагается разнообразие инструментальных средств, и их выбор зависит, в первую очередь, от решаемых задач. Для сайтов, ориентированных на мониторинг и управление технологическими процессами, предлагаются пакеты Microsoft InterDev или FrontPage. Пакет FrontPage используется:

- как визуальное средство, позволяющее «непрограммистам» реализовать Web-публикацию в среде клиент/сервер;
- для обслуживания Web-сервера и Web-сайтов на этом сервере;
- для создания Web-страниц. Web-страница с FrontPage поставляется с 16- и 32-разрядными версиями собственного сервера Personal Web Server, который может использоваться с ОС Windows 3.11/95/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 (приложение 7.5).

Приложение 7.5

CGI (Common Gateway Interface) – это стандартный шлюзовой интерфейс для запуска внешних программ под управлением сервера Web. Соответственно, приложениями CGI называются программы, которые, пользуясь этим интерфейсом, получают через протокол HTTP информацию от удаленного пользователя, обрабатывают ее и возвращают результат обработки обратно в виде ссылки на уже существующий документ HTML, другой объект или в виде документа HTML, созданного динамически. Приложения CGI позволяют также организовать связь между документами HTML и системами управления базами данных.

Сервер, содержащий, наряду со статическими, и динамические документы, называют активным интернет-сервером. Активные серверы создаются с использованием программных расширений CGI, ISAPI (приложение 7.6).

Приложение 7.6

Расширения ISAPI по своему назначению напоминают программы CGI. Но в отличие от последних эти расширения выполнены в виде библиотек динамической компоновки DLL, что имеет ряд преимуществ. Расширения ISAPI также получают данные от навигатора, обрабатывают их и посылают навигатору отчет. Однако, вместо получения данных в виде переменных среды и стандартного потока ввода STDIN, расширения ISAPI получают данные при помощи специально предназначенных для этого функций. Аналогично, вместо записи выходных данных в стандартный поток вывода (схема, работающая в CGI) расширение ISAPI вызывает специальную функцию.

Текстовые файлы страниц готовятся с использованием специального языка разметки гипертекста **HTML** (**H**yper**T**ext **M**arkup **L**anguage). Взаимодействие пользователя с

сервером WWW осуществляется через запросные 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).

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 другую нестандартную 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 этой угрозы Microsoft 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 соглашается на загрузку и запуск ActiveX, если нет – отказывается.

Язык HTML допускает использование языков программирования Java, JavaScript и VBScript. Java разработан фирмой Sun на основе языка Oak как платформно-независимый интерпретируемый объектно-ориентированный язык. Создание программ Java и размещение ссылок на них производится 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 одного сервера WWW могут ссылаться на страницы, расположенные и на других серверах в сети Интернет, включая серверы FTP, Gopher, конференции, электронные почтовые адреса.

Следует сказать несколько слов и о языке **XML** (e**X**tensible **M**ark-up-**L**anguage), у которого имеется общий «предок» с HTML, – стандартный обобщенный язык описания доку-

ментов **SGML** (**S**tandard **G**eneralized **M**ark-up **L**anguage). Язык XML имеет более строгий синтаксис. Прослеживается и тенденция: применение языка HTML 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, можно создать собственные, ориентированные на решение своих 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 представляет собой 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. 7.9).

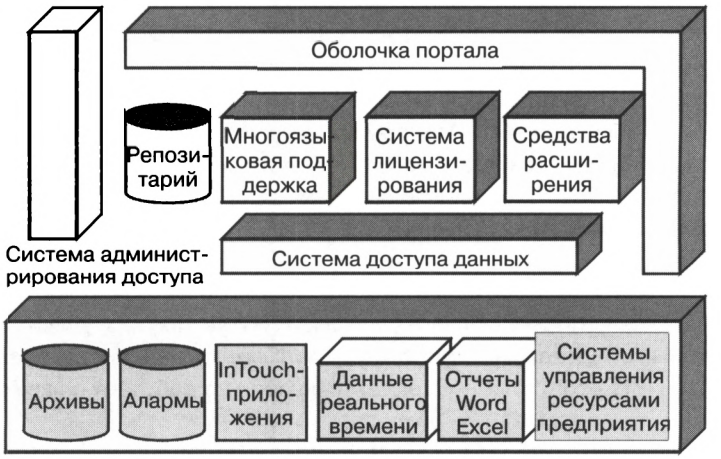


Рис. 7.9. *Структура портала SuiteVoyager*

Пакет предоставляет пользователю набор средств для просмотра и 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-формат (рис. 7.10).

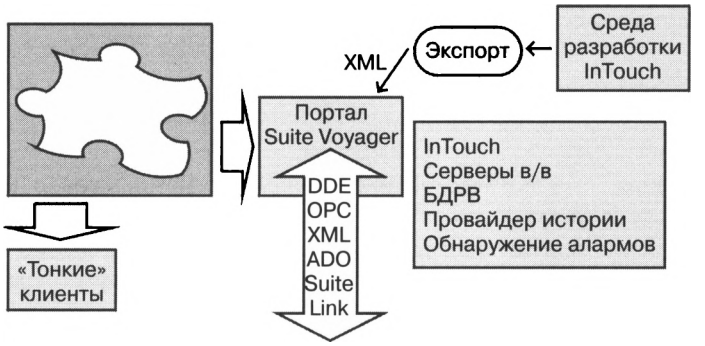


Рис. 7.10. *Решение на основе SuiteVoyager*

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 новые made-for-the-Web-технологии, такие, как XML.

7.4. Internet/Intranet решения от Citect

Пакет Plant2Business от Citect – 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 итоге 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 для хранения 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 системы управления, так и переданы в нее.

Plant2Net обеспечивает передачу данных из Plant2Business-сервера Internet/Intranet-клиентам по технологии «тонкого» клиента. Причем выбираются только необходимые в данный момент данные в виде иерархической структуры.

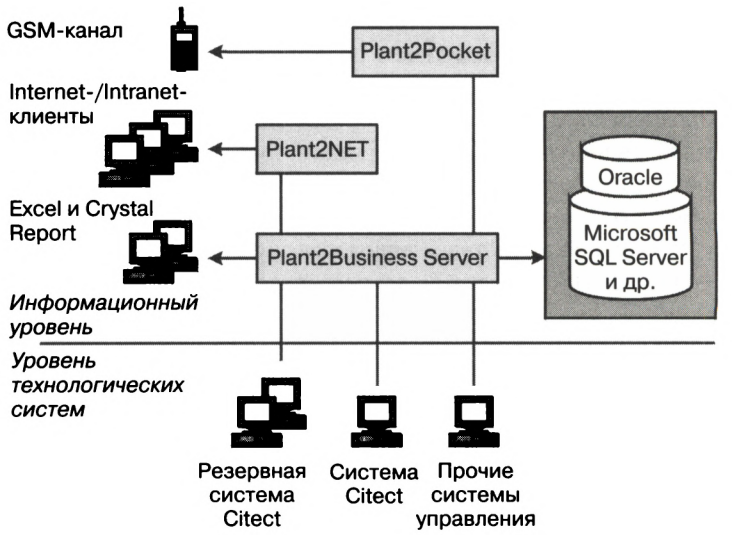


Рис. 7.11. *Схема информационных потоков*

На рис. 7.11 показана схема информационных потоков предприятия. 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.

Предлагаемый пакет обеспечивает возможность осуществлять в реальном времени не только сбор данных, аварийных сообщений с различных подсистем и ведение архивов,

но и доступ клиентских приложений к этой информации, в том числе по протоколам http/https.

FTP-клиенты. Citect поставляет и еще одно новое 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 заинтересованным клиентам.

Типичная реализация «толстого» или «богатого» клиента часто связана с расширением числа протоколов, которые поддерживают приложения SCADA. С точки зрения пользователя, необходимо просто приобрести лицензию исполняющей системы и использовать приложение SCADA как Internet/Intranet-клиента.

Применяются два типа «бедных» клиентов – терминал/серверные и Internet-клиенты (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.

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: 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 имеются различия в их реализации:

- SuiteVoyager как Web-серверное расширение обеспечивает получение информации из различных источников реального времени, базы данных реального времени IndustrialSQL Server и предоставление ее Internet-клиентам. Citect подчеркивает значимость сервера Plant2Business как базы данных, регистрируемых со всех источников информации;

- Citect предлагает протоколы для обмена как по локальной сети (TCP/IP, NetBEUI), так и по глобальной сети (FTP-протокол).

ГЛАВА 8. Заключение

В последнее десятилетие на многих предприятиях уже созданы и успешно эксплуатируются современные автоматизированные системы управления технологическими процессами (АСУТП), автоматизированные системы управления предприятием (АСУП), учётные системы. Безусловно, достигнуты большие успехи, но уже на стыке веков все чаще и чаще в специальной прессе стала подниматься проблема достаточно низкой эффективности внедренных систем. Оказалось, что созданные на разной программно-аппаратной платформе, хотя и современной, эти системы мало взаимодействуют между собой. Какими бы функциональными особенностями ни обладали указанные локальные системы, они не включены в единый производственный цикл и не образуют комплексную систему автоматизации производства.

SCADA-системы в подавляющем большинстве ответственны лишь за уровень промышленной автоматизации, обеспечивающий, главным образом, получение данных от различных датчиков и устройств ввода/вывода, представление собранной информации и её архивацию. Доступ же к этой информации со стороны руководителя предприятия, а также руководителей экономических подразделений до недавнего времени был лишь опосредованным.

Для анализа производства в целом, для моделирования его отдельных этапов, для выявления критических участков и слабых звеньев важна организация доступа к данным, отражающим весь процесс производства, с возможностью воздействия на него, в том числе и в реальном времени.

Попытки создания комплексных программных пакетов для реализации подобных задач автоматизации промышленных предприятий предпринимают компании-разработчики трёх групп:

- разработчики SCADA-систем;
- разработчики **ERP**-систем (**E**nterprise **R**esource **P**lanning);
- разработчики **MES**-систем (**M**anufactoring **E**xecutive **S**ystems).

8.1. О средствах, расширяющих возможности обработки данных

Разработчики SCADA-систем первыми осознали необходимость предоставления данных на все уровни управления предприятием, зрелость объективных условий (сетевые коммуникации, уровень аппаратных и программных средств и др.) и стали предлагать различные решения, расширяющие возможности обработки данных.

8.1.1. Отдельные опции и пакеты, расширяющие возможности обработки данных различного назначения

В целом все данные, собираемые с локальных систем, можно разделить на 3 группы:

1). **Технологические данные.** Используются операторами установок и диспетчерами в реальном времени. На практике функциональность, которая предлагается традиционными SCADA-пакетами, достаточна для реализации требований операторов, технологов;

2). **Диагностические данные.** Данные такого типа пока используются мало, но иногда регистрируются (на всякий случай). Основное их назначение – оценить состояние оборудования, которое обеспечивает производственный процесс. Значительная часть разработчиков SCADA-пакетов предлагает отдельные приложения, предназначенные для обнаружения ситуаций, способных привести к выходу оборудования из строя, простоев. Информацию эти системы получают в режиме реального времени из SCADA-приложений или напрямую из серверов ввода/вывода. Вся информация архивируется и впоследствии может использоваться для анализа. Класс программного обеспечения серии Downtime, ориентированный на обработку диагностических данных, имеет в своём составе средства для отображения информации о состоянии оборудования и о проблемах, возникавших на протяжении заданного периода времени. Для хранения конфигурационных данных и накопления архива используются обычные реляционные базы данных, такие, как MS SQL Server, Oracle. Важно отметить, что вся отчётная информация (как оперативная, так и архивная) может публиковаться на внутризаводском Web-сервере. В этом случае доступ к информации возможен с любого компьютера в заводской сети, на котором установлен браузер Internet Explorer. Таким образом, к достоинствам ПО класса Downtime можно отнести то, что оно позволяет:

- осуществлять мониторинг производства в реальном времени;
- обоснованно обновлять оборудование;
- получать самую точную информацию о простоях оборудования;
- снижать затраты на сбор информации;
- оптимизировать интервалы техобслуживания;
- автоматически или вручную регистрировать останов оборудования;
- настраивать источники информации с помощью удобного инструментария;
- просматривать информацию без необходимости иметь SCADA-систему;
- просматривать отчёты Internet;
- корректировать данные во время работы системы;
- архивировать информацию.

Однако для организации полноценного управления производственными фондами, куда входит управление техобслуживанием, складами, планово-предупредительными ремонтами (ППР) и снабжением, возможности этой системы недостаточны. Программное обеспечение серии Downtime является только промежуточным решением, так как позволяет получить оперативную информацию о техническом состоянии оборудования. Отдельному рассмотрению (за пределами книги)

подлежат системы управления производственными фондами, или **EAM**-системы (**E**nterprise **A**ssets **M**anagement **S**ystems);

3). **Хозрасчётные данные.** Используются для оценки товарно-сырьевого потоков, формирования балансов. Часто эти данные являются неполными.

Первичные данные (особенно хозрасчётные) нуждаются в обработке и согласовании. Поэтому с целью повышения качества SCADA-пакетов некоторые разработчики предлагают специальные программные опции, такие, как «статистическая обработка данных» (**SPC** – **S**tatistical **P**rocess **C**ontrol) и «аналитическая обработка данных» (например, QI Analyst от компании Wonderware или Quality от компании Citect). Задача согласования данных (data reconciliation) решается программными продуктами уровня MES.

8.1.2. Отдельные приложения, ориентированные на различные области применений

Перечень основных программных компонентов, предлагаемых разработчиками SCADA-пакетов в дополнение к своему базовому продукту, составлен на примере продуктов компаний Wonderware и Citect и включает следующие основные программные пакеты, которые хорошо интегрируются друг с другом:

- **IndustrialSQL Server** (Wonderware), **Plant2Business Server** (Citect) – базы данных реального времени (БДРВ) для внутризаводского применения. БДРВ позволяет накапливать и хранить «историческую информацию» о производственном процессе, собирая её с нескольких сотен устройств ввода/вывода и управления, а также большого количества SCADA-узлов. БДРВ объединяет эту информацию с данными о конфигурации, об аварийных ситуациях и событиях, с итоговыми и статистическими данными, с «историей» рецептов, с данными о ходе производства;

- **InTrack** (Wonderware), **Trace** (Citect) – системы слежения за процессом производства. Они позволяют наблюдать и отслеживать в реальном времени незавершённое производство, материально-технические запасы, использование оборудования, простои и т.д. Системы позволяют определять и моделировать производственные процессы, контролировать исполнение заказов на продукцию. Значительная часть внедрений данного класса продуктов выполнена для дискретного производства. Ниже описаны основные функции таких систем;

- **InBatch** (Wonderware), **Batch** (Citect) – системы гибкого управления процессами дозирования и смешивания. При помощи такого ПО пользователи в металлургической, химической, пищевой промышленности могут моделировать свои процессы, создавать рецепты, имитировать исполнение рецептов, сопоставляя их с моделью, управлять реальным процессом, пользуясь моделью для справок.

Компании Wonderware и Citect предлагают пользователям не только отдельные приложения, но также интегрированные среды, с помощью которых можно разрабатывать и конфигурировать комплексную распределённую систему автоматизации в целом.

8.2. Программные продукты от разработчиков SCADA-систем

8.2.1. Системы слежения за процессом производства (InTrack, Tracing)

Для осуществления оперативного контроля необходима информационная и коммуникационная среда, отражающая производственный процесс предприятия. Эта среда не только обеспечивает передачу данных с технологического уровня в финансовые модули и модули логистики, но и, что, пожалуй, самое главное, позволяет управлять процессом производства более гибко, уменьшая стоимость производства, увеличивая прибыль.

Конечно, никогда все процессы производства и бизнес-процессы не станут полностью автоматическими. Периодический анализ достоверной информации должен сопровождаться разработкой корректирующих мероприятий по ряду направлений, включая изменения регламентов производственного процесса, изменения, относящиеся к персоналу (обучение персонала, повышение квалификации и др.), изменения в инфраструктуре. И качество этих решений определяет эффективность производства в целом.

Функциональные возможности

Приложения, подобные InTrack или Trace, поддерживают следующие функции:

- слежение за материалами производственного процесса от сырья до готового продукта и управление ими;
- отслеживание состояния складов, диспетчеризация преимущественно дискретного производственного процесса;
- управление данными качества (поддерживает практику хорошего производства), управление инструкциями по проведению работ, определение стандартных процедур отдельных операций, сбор данных о потреблении материалов;
- генерация набора предопределенных отчётов, общий доступ к БД;
- мониторинг производства в реальном времени;
- слежение за ресурсами и управление – биллинг ресурсов, машин, труда.

Слежение – это мониторинг, регистрация изменения статуса и количества материалов, используемых в процессе производства и размещённых на складах.

Таким образом, в приложениях InTrack и Trace реализованы три взаимосвязанных процесса: контроль процесса производства, слежение за материалами и контроль технологических параметров.

Контроль процесса производства и история процесса. Процесс производства определяет ресурсы, используемые для производства продукта. Это операции, станки, операторы, документы, в том числе спецификации и другие необходимые для выпол-

нения операций нормативные документы. Хранилищем информации о ресурсах для оценки и будущего планирования является производственный архив.

Слежение за материалами и архив. В процессе контроля материалов отслеживаются материалы и продукты, которые используются для производства полуфабрикатов или продуктов. Процедура контроля материалов включает потребление и ограничение на материалы в процессе производства. В архиве регистрируются реальное потребление материалов, хранение, передвижение и производство материалов.

Контроль технологических параметров. Управление технологическими процессами гарантирует, что значения технологических параметров находятся в заданных пределах. При этом используются значения параметров потребления материальных и энергетических ресурсов, важных качественных показателей, параметров состояния оборудования и т. д.

В зависимости от отрасли весовые коэффициенты применения каждой из перечисленных информационных компонент меняются. Так, для фармацевтической промышленности важно отслеживать материалы и производимые товары.

Процесс моделирования

Все операции производства последовательно описываются разрабатываемой моделью процесса. Модель определяет уровень детальности процесса для регистрации в базе данных. Модель может не сопровождаться уровнем детальности, соответствующим реальному производству.

При моделировании используется понятие маршрута. В классической модели маршрут определяется как последовательность операций, которые являются прямым отображением основных этапов производства продукции (рис. 8.1).

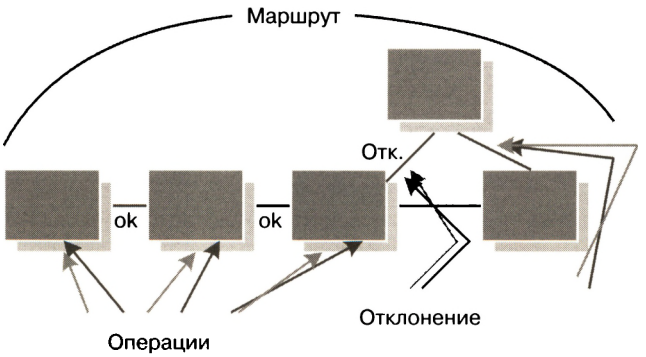


Рис. 8.1. *Модель процесса*

Поскольку продукты движутся по отдельным этапам маршрута, то перед каждым этапом проверяется наличие необходимого набора ресурсов для выполнения следующей операции. Предлагаемая концепция модели удобна для дискретного производства.

Для других типов производства классическая модель не является столь подходящей.

Для некоторых производств материалы перемещаются по маршруту быстро, поэтому слежение не может быть реализовано по каждой операции. Модель маршрута может только регистрировать точки, в которых материалы вводятся или удаляются из процесса производства, отслеживать количественные параметры материалов, передвижение материалов, субпродуктов, продуктов и т.п.

Для таких процессов потребление материалов может быть непрерывным. При этом быстроменяющиеся параметры потребления материалов рекомендуется сохранять в базе данных реального времени (БДРВ). Используя наборы данных в БДРВ, приложение может управлять большими объёмами информации для наиболее эффективного использования системных ресурсов.

Основные объекты. Модель производственного процесса состоит из структурных и динамических объектов. Структурные объекты – это более или менее статичные компоненты производства, такие, как операции. Одна и та же операция может использоваться на различных маршрутах при производстве различной продукции. Динамические объекты – это меняющиеся в процессе производства компоненты, прежде всего потребляемые в процессе производства материалы, называемые лотами.

Особенности и достоинства

Особенность рассматриваемых приложений InTrack и Trace заключается в том, что они обеспечивают достоверную и оперативную информацию о состоянии производственного процесса.

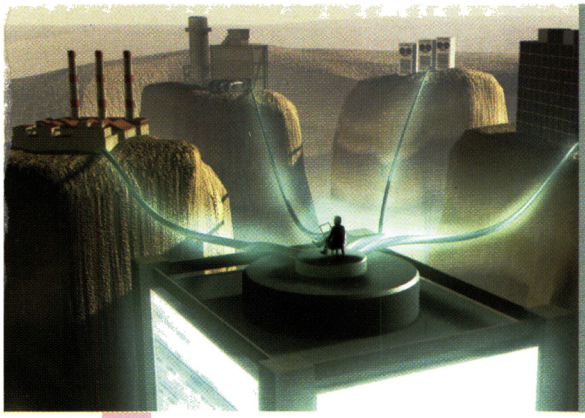
Во-первых, такая система позволяет связывать все аспекты производственного процесса вместе, вести учёт расходования сырья и полуфабрикатов, отслеживать процесс производства с учётом загруженности оборудования и проводить текущий контроль качества.

Во-вторых, персонал производственных и плановых управлений и отделов получает ясное представление о течении производственного процесса, локализует проблемные места, включая незавершённые маршруты, уровень запасов сырья и комплектующих.

В-третьих, такая система позволяет повысить качество обслуживания клиентов с помощью быстрого реагирования на запросы и отслеживания заказов, точного выполнения плановых заданий.

В-четвёртых, система просто интегрируется с другими приложениями комплексной информационной системы предприятия, поскольку поддерживает открытые стандартизованные интерфейсы, имея прежде всего готовое проработанное решение по обмену данными со SCADA-приложениями и БДРВ.

В-пятых, допускается выбор платформы для сервера базы данных, включая Oracle, MS SQL Server на серверах Sun и Microsoft (ОС Windows).



Citect

для высоконадежных применений

- **CitectSCADA - ПО для систем диспетчерского управления и сбора данных**
- **Plant2Business Solutions - база данных реального времени**
- **Citect IIM - решение для оперативного управления производством**
- **CitectSCADA Pocket - ПО для Pocket PC (удаленный контроль и управление производством)**
- **CitectSCADA Facilities - ПО для управления системами жизнеобеспечения зданий и сооружений**
- **CitectHMI - реализация человеко-машинного интерфейса (HMI) во встроенных системах серийных поставщиков изделий**

Бесплатная среда разработки

Масштабируемость и гибкость

Высокая производительность

Горячее резервирование

Развитый язык программирования

Встроенный генератор отчетов

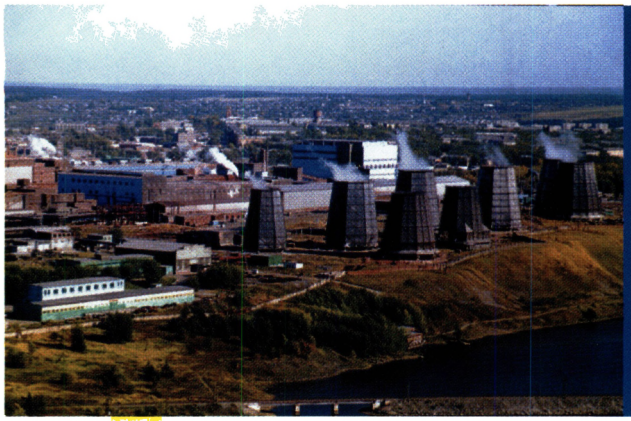
Многоязыковые проекты

- **Энергетика**
- **Химическая промышленность**
- **Пищевая промышленность**
- **Системы жизнеобеспечения зданий и сооружений**
- **Автомобильная промышленность**
- **Нефтегазовая промышленность**

ЗАО "РТСофт"
Москва, ул.Никитинская, 3
тел.: (095) 742 68 28, 967 15 05
факс: (095) 742 68 29
e-mail: rtsoft@rtsoft.ru

www.rtsoft.ru

RTSoft
СРЕДСТВА и СИСТЕМЫ АВТОМАТИЗАЦИИ



Factory Suite

для решения задач промышленной автоматизации

- **Сервер приложений для промышленной автоматизации (Industrial Application Server)**
- **Организация человеко-машинных интерфейсов (InTouch)**
- **База данных реального времени (IndustrialSQL Server)**
- **Программное обеспечение для статистического контроля процесса (QI Analyst)**
- **Программирование контроллеров на языках МЭК 1131-3 (InControl)**
- **Web-портал для доступа к данным о производстве через Internet (SuiteVoyager)**
- **Отслеживание и анализ простоев оборудования (DT Analyst)**

Анимирование объектов

Мастер-объекты, ActiveX, функции пользователя

Тренды реального времени и архивные

Многозадачность

Распределенные подсистемы алармов и архивов

Язык скриптов

Протоколы обмена

Около 800 I/O Servers, OPC-Servers

SPC, Recipe, SQL

- **Энергетика**
- **Химическая промышленность**
- **Пищевая промышленность**
- **Системы жизнеобеспечения зданий и сооружений**
- **Автомобильная промышленность**
- **Нефтегазовая промышленность**

ЗАО "РТСофт"
Москва, ул.Никитинская, 3
тел.: (095) 742 68 28, 967 15 05
факс: (095) 742 68 29
e-mail: rtsoft@rtsoft.ru
www.rtsoft.ru

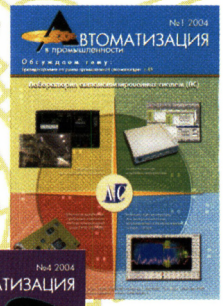
RTSoft
СРЕДСТВА и СИСТЕМЫ АВТОМАТИЗАЦИИ

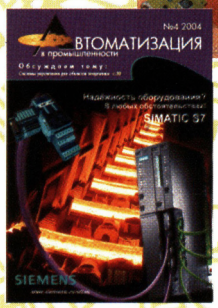
Ежемесячный научно-технический
и производственный журнал

АВТОМАТИЗАЦИЯ в промышленности

Журнал ориентирован на специалистов по промышленной автоматизации

Цель журнала: посредством оперативной, достоверной и независимой информации помочь специалистам ориентироваться в многообразии отечественных и зарубежных фирм, работающих в России, в номенклатуре продукции, новых технических решениях и концепциях, предлагаемых ими.





В журнале публикуются концептуальные, научно-практические и внедренческие статьи по разделам:

Производственные автоматизированные системы,
Системы управления бизнес-процессами,
Программное и алгоритмическое обеспечение,
Технические средства автоматизации

Раздел ***Обсуждаем тему...*** приглашает выступить специалистов, имеющих реальный опыт в обсуждаемой области

Автоматизация за рубежом - информация о состоянии и перспективах развития зарубежного рынка средств и систем автоматизации

Клуб журнала - консультации опытных специалистов, интервью, исторические очерки

Применение средств автоматизации - результаты промышленных внедрений

Большая роль отводится рубрикам, оперативно отражающим действительность российского рынка промышленной автоматизации: ***Новости, События, Фирмы***

Сайт журнала www.avtprom.ru

Новости журнала
Содержания вышедших номеров
Архивы журналов №№ 1-6 за 2003 г.
Новости промышленной автоматизации
Календарь предстоящих мероприятий
Правила оформления статей и рекламы
И многое, многое другое

Подписка оформляется:

В любом почтовом отделении
Подписные индексы в каталогах:
"Роспечать" **81874** "Пресса России" **39206**
В редакции или
Сети Интернет по адресу: **www.avtprom.ru**

Адрес редакции 117997, Москва, ул. Профсоюзная, 65, оф. 360
Телефон/факс (095) 334-91-30, телефон 315-19-55.
E-mail:avtprom@ipu.rssi.ru Http://www.avtprom.ru

ООО ИЗДАТЕЛЬСТВО "НАУЧТЕХЛИТИЗДАТ"

В журнале "Промышленные АСУ и контроллеры" публикуется самая свежая информация о состоянии рынка отечественных и зарубежных программных и технических средств АСУ и широко освещаются последние разработки ведущих фирм-представителей мирового рынка. Актуальность журнала также определяется серьезным анализом состояния и тенденций развития средств автоматизации и управления и описанием конкретных результатов внедрения этих систем на российских предприятиях.

Тел. 323-94-77, 324-45-85.
Тел./факс 323-90-10.
E-mail: asu@asucontrol.ru
pasuk@tgizdat.ru
pasuk@newmail.ru
http://www.asucontrol.ru

Подписные индексы:
в каталоге Агентства "Роспечать" 79216
в объединенном каталоге
"Пресса России" 40727
в каталоге "Издания органов
научно-технической информации
Агентства "Роспечать" 66731

При отсутствии систем данного класса все меры, принимаемые для улучшения процесса производства:

- не могут быть эффективными, поскольку не основаны на достоверной информации;
- запаздывают, поскольку отсутствует оперативное управление;
- приводят к усложнению системы, так как для решения частных временных проблем создаются постоянные структуры.

8.2.2. Системы управления непрерывным производством с элементами дозирования и смешивания (InBatch, Batch)

К основным функциональным возможностям приложений InBatch, Batch относятся моделирование процессов, создание рецептов, имитация исполнения рецептов, сопоставление их со специальной справочной моделью.

Поэтапное управление производственным процессом заложено в саму его модель через функции краткосрочного планирования, инициализации партий конечного продукта, управления циклами и этапами периодического процесса, статистики и учёта.

Приложения InBatch и Batch хорошо интегрированы со SCADA-приложениями, что позволяет оператору (диспетчеру) иметь интерактивный графический интерфейс для мониторинга и контроля периодических процессов.

Особенности

- **Управление периодическим процессом.** Поэтапное управление периодическим процессом заложено в модель процесса, что исключает необходимость написания собственного программного кода и резко снижает объём инженерных работ по сопровождению системы при эксплуатации. К конкретным функциям относятся: краткосрочное планирование, инициализация партий, управление циклами и этапами периодического процесса, статистика периодических процессов и составление отчётов.
- **Удобный графический интерфейс пользователя.** Технологи могут разрабатывать рецепты, используя удобный графический интерфейс пользователя, что ускоряет поступление готовых продуктов на рынок.
- **Моделирование процесса.** Созданная в приложении технологическая и информационная модель описывает возможности оборудования и технологических процессов, состав реквизитов, наиболее полно удовлетворяющих потребности специалистов предприятия в информации. Модель позволяет быстро изменять конфигурацию технологического процесса и системы управления при переходе на другой вид продукции.
- **Соблюдение регламента.** Рецепты в рассматриваемых приложениях создаются путём выборки возможностей оборудования и технологических процессов

предприятия из моделей процессов. Рецепты вводятся в виде Основных Рецептов (не зависят от оборудования и технологии) и преобразуются в Контрольные Рецепты (зависят от оборудования) во время выполнения программы. Управление рецептами распространяется также на потребности в оборудовании, ввод материалов, вывод материалов и процедуры.

- **Открытая архитектура и расширяемость** обеспечиваются набором API-функций для интеграции с внешними приложениями, такими, как ERP- или планирующие системы, которые позволяют проводить обмен формулы/рецепты, планировать сроки, материалы и производственные результаты. Эти функции упрощают интеграцию с ERP- и планирующими системами.

В таблице 8.1 классифицированы компоненты, предоставляемые разработчиками SCADA.

Название продукта	Функции	Краткое описание
SCADA	1. Стандартные функции	Программные средства создания и ведения рецептов (Recipe)
	2. Опции (дополнительные функции SCADA-систем)	Статистическая обработка первичных данных (SPC – Statistical Process Control)
Дополняющие SCADA-систему приложения	1. Продвинутая статистическая обработка	Программные средства класса Analyst расширяют функции SPC и SPC.PRO
	2. Учет простоев оборудования	На основе диагностических данных из SCADA-приложений программные средства класса Downtime консолидируют и позволяют анализировать информацию о состоянии и простоях оборудования
Базы данных реального времени	1. Специальный тип программного обеспечения, предназначенный для регистрации данных реального времени	БДРВ позволяет регистрировать и архивировать данные в реальном времени. С точки зрения доступа и использования технологических данных обеспечивается доступ по SQL-запросам. Практически всегда снабжается набором клиентских приложений
	2. Набор клиентских приложений	Уже устоявшийся набор приложений для просмотра динамически выбираемого набора переменных в виде трендов, таблиц, графиков зависимостей одной переменной

		от другой, обзорного приложения, позволяющего просматривать мнемосхемы процессов
Приложения для слежения за процессом производства	Реализует MES-функцию слежения за производственным процессом	Программы класса Track или Trace
Приложения управления непрерывным производством с элементами дозирования и смешивания	Реализует MES-функцию слежения за производственным процессом	Программы класса Batch
Программные средства подготовки отчетов	Функция документооборота	Программы класса Reporting обеспечивают генерацию документов в рамках обслуживаемого процесса

Таблица 8.1

8.2.3. Программные средства от поставщиков MES-систем

Системы управления производством, или MES-системы, являются информационной и коммуникационной средой, отражающей процесс производства на предприятии. Эти системы не только обеспечивают передачу данных с технологического уровня – уровня управления в реальном времени – в модули систем планирования ресурсов предприятия (АСУП), но и, самое главное, позволяют управлять процессом производства более гибко, уменьшая стоимость производства и увеличивая прибыль.

В руках персонала системы автоматизации производства являются инструментом управления и оптимизации. Это означает, что, используя информацию, в том числе и информацию реального времени, люди должны принимать решения о том, что и как делать в будущем. А качество этих решений определяет эффективность работы производства в целом.

В зависимости от круга задач, решаемых на уровне управления производством, эти системы можно классифицировать. И каждый класс систем обеспечивается специализированными программными продуктами, достаточно широко представленными на рынке средств автоматизации.

Лабораторные системы обеспечения качества продукции (**LIMS** – **L**aboratory **I**nformation **M**anagement **S**ystems). Они используются в химических лабораториях на предприятии и обеспечивают:

- регистрацию поступающих в лабораторию образцов (ресурсы, полуфабрикаты, товарная продукция);

- контроль и прохождение образцов через лабораторию;

- регистрацию результатов тестов и испытаний с учётом применения соответствующих методик расчёта и анализа;
- ввод информации о результатах тестирования образцов непосредственно с приборов;
- анализ результатов с использованием статистических методов;
- накопление различного рода статистических данных о технологическом процессе;
- выпуск различного рода отчётов и т.п.

Системы согласования данных, учёта потерь, сведения массовых, энергетических, тепловых, объёмных и качественных балансов. Ресурсные данные, регистрируемые на технологическом уровне, в основном используются в системе балансового расчёта. На основе данных о состоянии технологического процесса и потребляемых ресурсах:

- производится учёт продукции и расчёт периодических потерь предприятия;
- предоставляются точные, сбалансированные данные для инженерных расчётов и планирования;
- контролируются реальные потери;
- контролируется и корректируется процесс управления предприятием путём ежедневного расчёта массового баланса.

Необходимость рассчитываемого баланса может быть вызвана потерями двух типов:

- реальными потерями в процессе производства, включая отходы в брак, использование на собственные нужды и т.п.;
- потерями из-за неточности измерений, недостаточной точности датчиков, ошибки учёта продукта при погрузке-разгрузке в процессе транспортировки.

Таким образом, ошибки в измерениях параметров процессов являются причиной финансовых потерь. Ошибки калибровки, дефекты датчиков, неправильно регистрируемые измерения, ошибки измерений в резервуарах и неучтенные потери в ходе технологического процесса делают собранные о процессах данные несогласованными и, следовательно, некорректными для анализа (расчёта на их основе производительности производства, решения задач оптимизаций технологических процессов). Используя специальные алгоритмы, данная система рассчитывает согласованные измерения процесса, которые должны иметь минимальное отклонение от действительных значений. Одновременно должны сходиться и массовый, и энергетический балансы.

К числу коммерчески доступных программных продуктов, применимых для оценки потоков распределения и расчёта балансовой схемы, можно отнести программное

обеспечение Datacon компании Simulation Sciences Inc., ProductionBalance компании Honeywell.

Системы планирования и моделирования технологических процессов и планирования производственных процессов. Такая система объединяет модели отдельных установок в общую модель производства и осуществляет оптимизацию по модели. Особенность этих систем состоит в том, что они зависят от типа производства (нефтепереработка, нефтехимия, химия и т. д.). Этот класс систем позволяет осуществлять технологическое моделирование производств с использованием значительного числа логических установок, описывающих разные варианты их работы. При этом прослеживаются различные качественные показатели по всей схеме производства – от сырья до товарной продукции – с использованием нелинейных моделей процессов. В результате получается полная технико-экономическая картина производства: объём выпуска продукции по видам с качественными показателями и оптимальными целевыми рецептурами смешения, оптимальные загрузки установок, переменные составляющие затрат и другие характеристики.

Совокупность реализуемых каждой системой наборов функций меняется в зависимости от выбранного продукта. Поэтому при внедрении таких систем на предприятии определяется продукт, который соответствует типу производства, степени реализации каждой функции, степени самостоятельности каждой системы среди других и т.д.

Системы управления документооборотом. Такие системы обеспечивают регистрацию и формирование отчётов-форм с отслеживанием процесса производства во всех срезах, включая рабочие инструкции, рецепты, процедуры выполнения операций, межсменные рапорты и т.д. Системы подготовки отчётов предлагаются сейчас различными компаниями – от SCADA-разработчиков (поддержка отчётов в SuiteVoyager от Wonderware, Production Reporting от Citect) и разработчиков систем класса Workflow до поставщиков универсальных систем документооборота, таких, как NuGenesis от GE Electric.

Решение задач по автоматизации производства предполагается с помощью специализированных COTS-продуктов, реализованных в виде программных модулей.

Классы задач и программные модули:

1. Оперативно-календарное планирование (Preactor, Fobos, RPMS и др.);

2. Согласование данных,балансовые расчеты (DataCon, ProductionBalance и др.);

3. Моделирование работы отдельных установок (APC) и производственного процесса в целом.

Примечание: *Особенность этой группы систем состоит в том, что коммерческие продукты, поддерживающие их, ориентированы на определенную отрасль или отрасли.*

8.2.4. Интеграция с системами АСУП

До последнего времени две подсистемы автоматизации промышленных предприятий – АСУП и АСУТП – развивались обособленно и независимо друг от друга. Каналы обмена между этими подсистемами, особенно оперативные, оказались достаточно слабыми.

Возможно, так продолжалось бы и дальше, но необходимость автоматизированной доставки технологической информации, в том числе и оперативной, на более высокие по отношению к АСУТП уровни стала очевидной. Надо наконец избавиться от субъективного (человеческого) фактора, который зачастую мешает принятию взвешенного, единственно правильного решения.

АСУП представляет собой интегрированный программный комплекс, направленный на организацию управления и анализа финансово-хозяйственной деятельности предприятия, и состоит из ряда модулей:

- финансовый модуль – интегрирует потоки финансово-экономических данных;
- модуль сбыта – поддерживает систему регистрации заказов, функционирующую на базе данных о клиентах и контактах;
- модуль учёта основных средств – обеспечивает возможность постоянного наблюдения за калькуляционным и иным движением стоимости;
- модуль управления персоналом;
- модуль планирования и управления производством;
- модуль планирования и управления проектами;
- модуль технического обслуживания и ремонта оборудования;
- модуль управления материальными потоками. (Под материальными ресурсами подразумевается сырьё, готовая продукция, побочные продукты, складские запасы.);
- модуль расчёта затрат.

Следует отметить, что частично функции систем управления производством и АСУП пересекаются. Например, модуль управления товарно-сырьевыми потоками призван контролировать запасы на складах с целью обеспечения оптимального размера складских запасов.

В то же время система слежения за производственным процессом (InTrack или Trace) главной задачей считает отслеживание регламентов технологического процесса, а складские ресурсы необходимы для его обеспечения. Таким образом, для этих систем данные о складских запасах являются общим информационным ресурсом.

В настоящее время созрели объективные (техническая база) и субъективные (понимание руководства предприятия) условия для интеграции подсистем АСУП и АСУТП.

Необходимыми предпосылками для решения этих задач стали недавно появившиеся единые сетевые протоколы и современные информационные технологии. Ниже приводятся наиболее распространённые способы информационной интеграции:

- использование баз данных (БД), включая применение их в качестве буфера для обмена, позволяет обеспечивать оперативный обмен данными между подсистемами. Причём БД служат как основой функционирования самих подсистем, так и средством, используемым для хранения функциональных данных;

- применение класса продуктов, основным назначением которых является импортирование объектов из одной подсистемы и экспортирование их в другую. Например, в продукте VisualFlow компании EnvisionIt для интеграции с системами SAP R/3 определены следующие интерфейсы:

  - интерфейс PPPI;
  - поддержка PI-PCS BAPI;
  - сертифицированные SAP;
  - интерфейс общего назначения R/3 RFC;
  - Access 10,000+RFC;
  - CALL_TRANSACTION RFC;
  - поддержка IDOC;
  - поддержка пользовательского ABAP/4;
  - интерфейс баз данных;
  - доступ к таблице/полю;
  - поддержка всех R/3 баз данных;

- использование готовых решений для предприятий. Следует отметить процесс объединения компаний-разработчиков продуктов АСУП и АСУТП (например, Wonderware и Protean).

Каждый уровень программного обеспечения (АСУТП – системы управления производством – АСУП) решает свои специфические задачи (опираясь на данные более низкого уровня), ориентирован на соответствующий персонал предприятия и обеспечивается необходимой для этого уровня системой документооборота.

Интеграция информации трёх уровней позволяет обеспечить автоматический контроль за выполнением производственных и хозяйственных процессов в соответствии с заранее определёнными процедурами и правилами, позволяет формировать интегральные, объективные параметры производства – его ключевые показатели.

8.2.5. Ключевые показатели производства (KPI – Key Performance Indicator)

Руководители, вынужденные решать первоочередные бизнес-задачи и желающие непременно добиться успеха, стремятся найти и реализовать «правильные» методики,

на основе которых можно действительно добиться роста эффективности. Есть множество испытанных и хорошо проверенных технологий, начиная со сбалансированной системы показателей (Balanced Scorecard) и Six Sigma (приложение 8.1) и заканчивая современным стандартом ISO9001, основанным на принципе «непрерывного совершенствования» (Continuous Improvement). Какой бы метод ни был избран, первостепенную важность имеет видимость на каждом этапе процесса.

Приложение 8.1

Методология Six Sigma

«Сигма» – так называется буква греческого алфавита, которой в статистике традиционно обозначается «отклонение». Критерий «шесть сигма» (Six Sigma) применительно к нормальному распространению дефектов означает, что на миллион экземпляров продукта приходится 3,4 дефектных. Цель программы Six Sigma, по крайней мере в начале, состояла в том, чтобы минимизировать отклонения (или дефекты) при производстве продуктов.

Данную концепцию в 80-х годах предложила корпорация Motorola. Её инженеры пришли к выводу, что новые продукты, которые часто не оправдывают ожиданий пользователей, можно с самого начала производить без дефектов. Это была поистине революционная идея для промышленного производства – соотносить требования потребителей и производительность с теми показателями, которые заложены в продукт ещё в процессе производства, а не после него. Методология Six Sigma предполагает выполнение двенадцати шагов (см. табл.).

Двенадцать шагов методологии Six Sigma

Измерение	1. Выбор характеристик CTQ
	2. Определение стандартов работы
	3. Подтверждение системы измерения
Анализ	4. Установление возможностей продукта
	5. Определение целей с точки зрения производительности
	6. Выявление источников отклонения
Совершенствование	7. Просмотр потенциальных причин брака
	8. Выявление взаимоотношений переменных
	9. Установление операционных допусков
Контроль	10. Подтверждение системы измерения
	11. Определение возможностей процесса
	12. Реализация средств управления процессом

Традиционно видимость обеспечивалась с помощью так называемых ключевых показателей результативности (**K**ey **P**erformance **I**ndicator – **KPI**), которые представляются через специальную инструментальную среду и дают понимание эффективности различных мероприятий как в настоящее время, так и в ретроспективе. Особенностью ПО, ориентированного на расчёт ключевых показателей, является способность интегрироваться с разноуровневыми подсистемами предприятия, в том числе и по протоколам реального времени. Двумя ключевыми компонентами в комплексной

системе являются модель данных предприятия (Plant Data Model), которая управляет информацией, связанной с деятельностью предприятия, и модуль обмена данными (Data Exchange Module), который связывает различные подсистемы и приложения предприятия в единую инфраструктуру. Благодаря доступности информации, предоставляемой такой системой, обеспечиваются:

- просмотр данных о процессах, складских запасах и качестве продукции в реальном времени и исторически;
- принятие просчитанных решений в целях оптимизации выбора сырья, операционных режимов и соотношений продуктов с одновременным достижением более высокого качества получаемых продуктов;
- анализ по схеме «что, если» для оценки деловых возможностей.

Открытость и доступность информации для децентрализованного принятия решений

Сегодняшнему бизнесу информация о производственном процессе крайне необходима. Решения должны приниматься оперативно, и столь же оперативными обязаны быть ответные действия. Этого можно добиться, обеспечив выдачу KPI посредством таких механизмов, как электронные ситуационные инструментальные панели и средства визуального анализа. С помощью простого процесса «Определить, Измерить, Анализировать, Улучшить, Управлять» можно проиллюстрировать, как использование ключевых показателей результативности обеспечивает видимость процесса на этапах измерения, анализа и управления. Исходные данные поставляют SCADA-системы, системы управления производством и АСУП.

По большому счёту, есть два основных пути совершенствования производства. Один состоит в том, чтобы повышать его (производства) эффективность, т.е. выжимать из него как можно больше путём различных улучшений, касающихся использования времени, фондов, ресурсов и денег. Другой путь – добиваться большей результативности в смысле повышения качества и более точного соответствия спецификации заказчика.

Ситуационная панель или экран часто отображают следующие группы показателей:

1. Интегральные данные:
   - объём продукции;
   - количество продукции, изготовляемое в единицу времени;
2. Эффективность использования производственных фондов (пример в приложении 8.2);
3. Соответствие регламенту производственного процесса;
4. Параметры качества произведенной продукции;

5. Ретроспективные параметры, дающие представление об изменении характеристик с течением времени.

Фактически KPI-показатели – это набор формул, по которым показатели постоянно вычисляются и отражают состояние процесса в любой момент. В результате руководители и другие сотрудники получают возможность сравнивать эффективность своих активов беспристрастным и унифицированным образом.

Кроме того, такие системы снижают риск воздействия необъективной и/или подтасованной информации на принятие важнейших бизнес-решений. Дополнительным бонусом является возможность получать представление об истинном положении дел на предприятии и выявлять системные проблемы.

Приложение 8.2

Примеры ключевых параметров для главных специалистов предприятия

Отношение стоимости поддержки к бюджету:

% = Общая стоимость поддержки x 100/Общий бюджет
% = Общая стоимость трудовых ресурсов x 100/Бюджетная стоимость ресурсов
% = Общая стоимость материалов x 100/Бюджетная стоимость материалов

Отношения стоимостей поддержки:

% = Общая стоимость трудовых ресурсов x 100/Общая стоимость поддержки
% = Общая стоимость материалов x 100/Общая стоимость поддержки
% = Общая стоимость поддержки x 100/Общая стоимость продукта

Ситуационные панели настраиваются для разных классов пользователей. Ситуационная панель, реализованная с помощью SCADA-приложений, систем документооборота, позволяет извлечь максимум пользы. В качестве типичных конфигураций можно назвать такие, как:

- производственная панель, ориентированная на KPI с информацией о процессе изготовления продукции и материальных запасах;
- панель технического обслуживания, содержащая те ключевые показатели, которые имеют отношение к активам и техобслуживанию;
- панель качества, где представлены KPI качества продукции и лабораторных исследований;
- панель подразделения или предприятия, которая даёт сбалансированный обзор ключевых показателей результативности, касающихся производства, активов и качества;

- панель общего пользования, призванная создавать у работников положительную мотивацию соответствующими показателями.

Операционный бизнес-план содержит в себе ряд целей, затрагивающих вопросы производства, качества и затрат. Такие цели рассчитаны на обеспечение рентабельности предприятия и получение прибылей акционерами. Визуализация ключевых показателей результативности средствами электронных инструментальных панелей даёт доступ к этой информации всем сотрудникам компании. Приобщение работников к бизнес-целям предприятия делает эти цели общими и позволяет сконцентрировать усилия всего коллектива на основных направлениях. С показателями KPI руководители получают в свои руки мощный инструмент для создания мотивации на всех уровнях компании.

Привлечение ключевых показателей результативности для повышения эффективности предприятия отнюдь не ново. То, что обсуждается сейчас, – это автоматическая генерация KPI на основе комбинации данных реального времени и деловых операций. Надёжная, оперативная и согласованная выдача KPI-показателей позволяет сделать бизнес более динамичным и чутко реагирующим на нужды клиентов. Ситуационная панель образует своего рода концентратор бизнес-информации, способный получать данные от всех ключевых подсистем предприятия, преобразовывать эти данные при помощи делового регламента и представлять результат в виде оперативной KPI-информации.

Книгой о SCADA-системах авторы лишь немного приоткрыли дверь в интересный, поистине захватывающий мир технологической и производственной автоматизации. Сегодня уже можно с уверенностью сказать, что понятия SCADA-системы и АСУТП неразделимы. Но начавшаяся в конце XX века «головная боль» выбора SCADA-систем прошла далеко не сразу (да и прошла ли?). А «направлением главного удара» становятся программные продукты для MES-систем, рынок которых в России не так давно начал формироваться. И он будет развиваться с помощью как разработчиков SCADA, так и специализированных компаний.

Список литературы

1. Plessmann K., Wollert J., Kutsevich I. Kiryukhin S.,Pankrats E.,Vystavkin A. «A Modular, Multi-PC System for Real-Time Applications», The Tenh Internation Symposium on Problems of Modular Information Computer Systems and NetWorks, ESONE Committee, 1995.

2. Kutsevich I., Pertsovsky M. «User Environment of Acquisition, Processing for Stand Tests», Conference proceedings, RTD'95, ESONE Committee, Warsaw, Poland, pp.63-66, 1995.

3. Kutsevich I., Pertsovsky M. «Integrated Environment for Acquisition, Processing and Control by Digital Data», The Internation Symposium on Problems of Modular Information Computer Systems and NetWorks, ESONE Committee, 1996.

4. Куцевич Н.А. «SCADA-системы, или муки выбора», PCWeek, № 32, 1998.

5. Куцевич Н.А. «FactorySuite2000 – комплексный инструментарий следующего поколения», PCWeek, № 42, 1998.

6. Жданов А.А. «Еще раз о Hannover Messe'99», PCWeek, № 22, 1999.

7. Куцевич Н.А. «Программное обеспечение систем контроля и управления и Windows-технологии», Мир Компьютерной Автоматизации (МКА), № 3, 1999.

8. Куцевич Н.А. «Интегрированный пакет комплексной автоматизации FactorySuite 2000», МКА, № 3, 1999.

9. Куцевич Н.А. «Реляционные базы данных и IndustrialSQL Server – база данных реального времени», МКА, № 3, 1999.

10. Куцевич Н.А. «Новые технологии и MMI-системы», Открытые системы, № 4, 1999.

11. Куцевич Н.А. «FS2000 и автоматизация промышленного производства», Приборы и системы управления, № 5, 1999.

12. Куцевич Н.А. «SCADA-системы. Взгляд со стороны», PCWeek, № 33, 1999

13. Синенко О.В. «Открытые технологии автоматизации в российском ТЭК», Промышленные АСУ и контроллеры, № 5, с. 8-14, 1999.

14. Синенко О.В. «Об интеграции АСУП и АСУТП в единую систему», Промышленные АСУ и контроллеры, № 10, с. 3-7, 2000.

15. Куцевич Н.А. «SCADA-системы: проблемы тестирования», МКА, № 1, 2000.

16. Куцевич Н.А. «Об интеграции АСУТП и АСУП», МКА, № 3, 2000.

17. Куцевич Н.А. «Citect – новая SCADA-система на российском рынке и новые возможности», Промышленные контроллеры и АСУ, № 1, 2000.

18. Куцевич Н.А. «Citect – текущее состояние, ближайшие перспективы», Промышленные контроллеры и АСУ, № 6, 2000.

19. Куцевич Н.А. «Интернет-решения для продуктов Wonderware», Промышленные контроллеры и АСУ, № 9, 2000.

20. Куминов В., Куцевич Н. «Свидание назначено: интеллектуальное производство и бизнес встречаются во всемирной сети, в реальном времени», МКА, № 1-2, 2002.

21. Синенко О.В. «Анализ производственных процессов и подход к созданию комплексных систем управления производством», Нефтяное хозяйство, № 10, 2002.

22. Синенко О.В. «Оптимизация производственного управления», Экономика и жизнь, № 29, с. 20, 2002.

23. Синенко О.В. «Подход к анализу производственных процессов и созданию комплексных систем управления ресурсами», МКА, № 4, с. 14-21, 2002.

24. Куцевич И.В. «Моделирование и оптимизация в нефтехимическом производстве», МКА, № 4, 2003.

25. Синенко О.В. «Автоматизация диспетчерского контроля и управления потоками непрерывного производственного процесса как резерв повышения эффективности (на примере глиноземного производства)», Автоматизация в промышленности, № 6, с. 50, 2003.

26. Гребнев С.А., Кузякин В.И., Синенко О.В. «Интеграция АСУ: вчера, сегодня, завтра», Автоматизация в промышленности, № 9, с. 28, 2003.

27. Синенко О.В. «Новые возможности роста. Умное предприятие – это реально», Бизнес-Обозрение, № 2, с. 52, 2003.

28. Синенко О.В. «Создание комплексных систем управления производством», R-magazine, № 2, с. 52, 2003.

29. Синенко О.В., Кузякин В.И. «Современные подходы к интеграции АСУ», МКА, № 5, с.28, 2003.

30. Куцевич Н.А., Синенко О.В. «Ключевые показатели эффективности производства и их применение», Нефтяное хозяйство, с. 34-36, октябрь 2003.

Об авторах

Андреев Евгений Борисович, доцент кафедры АТП (автоматизация технологических процессов) Российского государственного университета нефти и газа им. И.М.Губкина.

Кандидат технических наук Куцевич Надежда Александровна, директор Центра SCADA-систем, интеграционных продуктов и услуг ЗАО «РТСофт», автор многих публикаций, посвященных SCADA-продуктам.

Кандидат технических наук Синенко Ольга Викторовна, генеральный директор ЗАО «РТСофт», автор ряда статей, посвященных современным подходам к созданию комплексных систем управления производством, а также вопросам интеграции АСУП и АСУТП.

Андреев Евгений Борисович
Куцевич Надежда Александровна
Синенко Ольга Викторовна

SCADA-системы: взгляд изнутри

Подготовка макета: *Нелли Рыбакова*
Дизайн обложки: *Светлана Романовская*
Координатор проекта *Дмитрий Вострикov*

Подписано в печать 03.06.2004.
Формат 70×100/16.
Печать офсетная.
Тираж 3000 экз.
Заказ № 707

ООО «РТ Софт»
105483, Москва, ул. Никитинская, д. 3, стр. 1.
Тел./факс: (095) 367-9036

Отпечатано с электронного носителя
в ГМП «Первая Образцовая типография».
115054, Москва, Валовая, 28