Тема 2. Пользовательский интерфейс, SCADA-пакеты

Функции SCADA. Разработка человеко-машинного интерфейса SCADA как система диспетчерского управления. SCADA как часть системы автоматического управления. Хранение истории процесса. Безопасность SCADA. Безопасность SCADA. Свойства SCADA. Инструментальные свойства. Эксплуатационные свойства. Степень открытости. Экономическая эффективность. Обзор программного обеспечения MasterSCADA, Vijeo Citect, WinCC.

Оглавление

Классификация программных средств систем управления технологическими процессами

В типовой архитектуре SCADA-системы явно просматриваются два уровня:

✓ ***уровень локальных контроллеров***, взаимодействующих с объектом управления посредством датчиков и исполнительных устройств;

✓ ***уровень оперативного управления*** технологическим процессом, основными компонентами которого являются серверы, рабочие станции операторов/диспетчеров, АРМ специалистов.

Каждый из этих уровней функционирует под управлением специализированного программного обеспечения (ПО). Разработка этого ПО или его выбор из предлагаемых в настоящее время на рынке программных средств зависит от многих факторов, прежде всего от решаемых на конкретном уровне задач.

Различают **базовое** и **прикладное** программное обеспечение (рис.2.1).

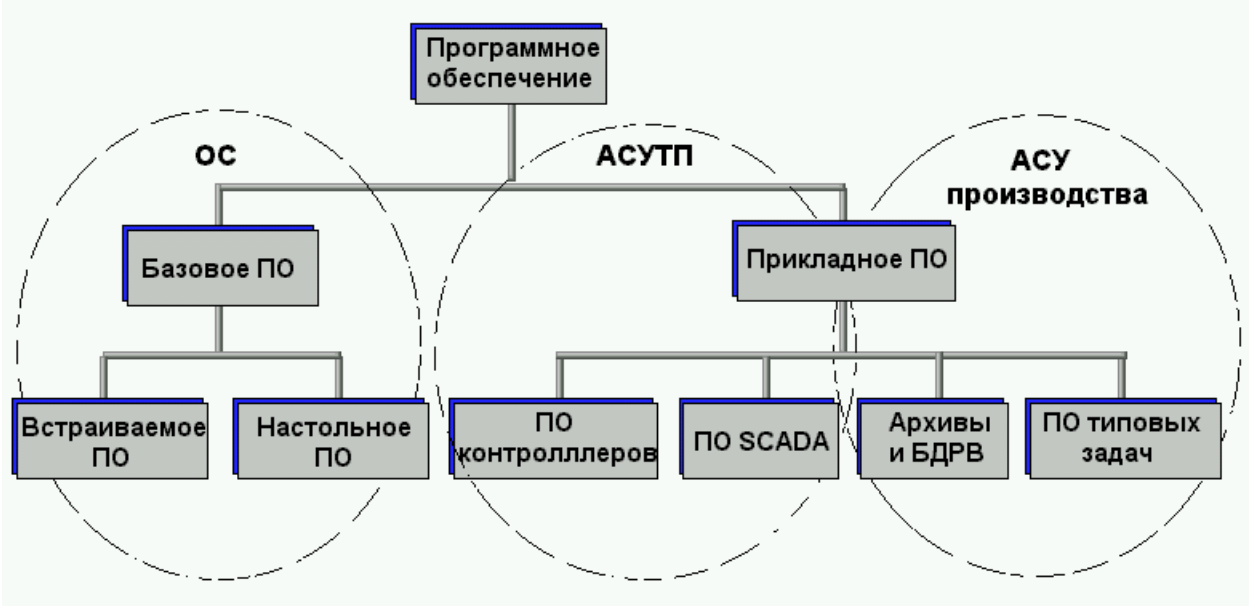

Рис. 2.1. Классификация программных средств системы управления.

➢ ***Базовое*** ПО включает в себя различные компоненты, но основным из них является операционная система (ОС) программно-технических средств АСУТП. Каждый уровень АСУТП представлен «своими» программно-техническими средствами: на нижнем уровне речь идет о контроллерах, тогда как основным техническим средством верхнего уровня является компьютер. В соответствии с этим в кругу специалистов появилась и такая классификация: **встраиваемое** и **настольное** программное обеспечение.

Очевидно, требования, предъявляемые к встраиваемому и настольному ПО, различны. Контроллер в системе управления наряду с функциями сбора информации решает задачи автоматического непрерывного или логического управления. В связи с этим к нему предъявляются жесткие требования по времени реакции на состояние объекта и выдачи управляющих воздействий на исполнительные устройства. Контроллер должен ***гарантированно*** откликаться на изменения состояния объекта за ***заданное*** время.

Для решения подобных задач рекомендуется применение **ОС реального времени** (ОСРВ). Такие операционные системы иногда называют детерминированными, подразумевая под этим гарантированный отклик за заданный промежуток времени. Большинство микропроцессорных устройств (в том числе контроллеры и компьютеры) используют механизм прерываний работы процессора. В ОС реального времени, в отличие от ОС общего назначения (не гарантирующих времени исполнения), прерываниям присвоены приоритеты, а сами прерывания обрабатываются за гарантированное время.

Операционная система **Windows** знакома всем как настольная система. Но это, прежде всего, относилось к платформам Windows 3.xx/95, в которых действительно отсутствует поддержка реального времени. Ситуация резко изменилась с появлением Windows NT. Сама по себе Windows NT не являлась операционной системой реального времени (ОС РВ) в силу ряда причин:

- ✓ поддержка аппаратных (а не программных) прерываний,
- ✓ не было приоритетной обработки отложенных процедур и др.

Первая ОС РВ была создана компанией VenturCom. Был разработан модуль Real Time Extension (RTX) - подсистема реального времени (РВ) для Windows NT. Эта подсистема имеет собственный планировщик со 128 приоритетами прерываний, который не зависит от NT. Максимальное время реакции на прерывание составляло 20-80 мкс вне зависимости от загрузки процессора. Теперь при каждом прерывании от таймера приоритет передавался критичным по времени задачам. А в оставшееся от их работы время выполнялись «медленные» процессы: ввод/вывод, работа с диском, сетью, графическим интерфейсом и т. п.

32-разрядная **Windows CE** была создана компанией Microsoft для малых компьютеров (калькуляторов), но в силу ряда достоинств стала претендовать на роль стандартной ОС реального времени. К числу этих достоинств относятся:

- ✓ открытость и простота стыковки с другими ОС семейства Windows;
- ✓ время реакции порядка 500 мкс;
- ✓ значительно меньшие по сравнению с другими ОС Windows требования к ресурсам памяти и возможность построения бездисковых систем.

А в 1999 году компанией Direct by Koyo ОС Windows CE была впервые установлена на платформу микроPLC.

Выбор операционной системы программно-технических средств **верхнего уровня** АСУТП определяется прикладной задачей (ОС общего пользования или ОСРВ). Но наибольшую популярность и распространение получили различные варианты ОС Windows Ими оснащены программно-технические средства верхнего уровня АСУТП, представленные персональными компьютерами (ПК) разной мощности и конфигурации - рабочие станции операторов/диспетчеров и специалистов, серверы баз данных (БД) и т. д.

Такая ситуация возникла в результате целого ряда причин и тенденций развития современных информационных и микропроцессорных технологий.

Вот несколько основных аргументов в пользу Windows:

- ✓ Windows имеет очень широкое распространение в мире, в том числе и в России, в связи с чем легко найти специалиста, который мог бы сопровождать системы на базе этой ОС;
- ✓ эта ОС имеет множество приложений, обеспечивающих решение различных задач обработки и представления информации;
- ✓ ОС Windows и Windows-приложения просты в освоении и обладают типовым интуитивно понятным интерфейсом;
- ✓ приложения, работающие под управлением Windows, поддерживают общедоступные стандарты обмена данными;
- ✓ системы на базе ОС Windows просты в эксплуатации и развитии, что делает их экономичными как с точки зрения поддержки, так и при поэтапном росте;
- ✓ Microsoft развивает информационные технологии (ИТ) для Windows высокими темпами, что позволяет компаниям, использующим эту платформу «идти в ногу со временем».

Также следует учитывать и то, что неотъемлемой частью верхнего уровня АСУ ТП является человек, время реакции которого на события недетерминировано и зачастую достаточно велико. Да и сама проблема реального времени на верхнем уровне не столь актуальна.

С точки зрения разработки системы управления предпочтительна такая программная архитектура, в которой ПО всех уровней управления реализовано в единой операционной системе. В этом случае «автоматически» снимаются все вопросы, связанные с вертикальным взаимодействием различных программных компонент системы управления. Но на практике это далеко не так. Достаточно часто в разрабатываемых системах контроля и управления нижний и верхний уровни реализуются в разных ОС. И наиболее характерна ситуация, когда на уровне контроллера используется ОС реального времени, а на уровне оператора/диспетчера SCADA-система функционирует под Windows. Без специализированных решений по организации взаимодействия между подсистемами здесь не обойтись.

Для функционирования системы управления необходим и еще один тип ПО - **прикладное программное обеспечение** (ППО).

Известны *два пути* разработки прикладного программного обеспечения систем управления:

1) создание собственного прикладного ПО с использованием средств традиционного программирования (стандартные языки программирования, средства отладки и т.д.);
2) использование для разработки прикладного ПО существующих (готовых) инструментальных средств.

Первый вариант является наиболее трудоемким. Применение языков высокого уровня требует соответствующей высокой квалификации разработчиков в теории и технологии программирования, знания особенностей конкретной ОС, тонкостей аппаратного обеспечения (контроллеров). Очевидно, что стоимость и время разработки ППО будут недопустимо велики.

Второй вариант является более предпочтительным, поскольку на сегодняшний день в мире уже создано несколько десятков инструментальных систем, хорошо поддерживаемых, развиваемых и нашедших применение при создании десятков и сотен тысяч проектов автоматизации. Эти проверенные временем программные средства упрощают (разработчики интерфейсов - не высококлассные программисты, а специалисты по автоматизации), ускоряют и значительно удешевляют процесс разработки.

С точки зрения области применения готовые инструментальные средства можно разделить на *два класса*:

1) средства, ориентированные на разработку программ управления внешними устройствами, контроллерами - **CASE**-системы (**Computer Aided Software Engineering**);
2) средства, ориентированные на обеспечение интерфейса оператора/ диспетчера с системой управления – **SCADA**-системы (**Supervisory Control And Data Acquisition** - диспетчерское управление и сбор данных).

Контроллеру требуется **программа**, в соответствии с которой он взаимодействует с объектом, выполняя следующие функции:

- ✓ сбора данных с объекта,
- ✓ логическое управление (например, выполнение блокировок).
- ✓ непрерывное управление отдельными параметрами или технологическим аппаратом (процессом) в целом.

Фирмы, производящие оборудование для построения систем автоматизации, всегда стремились сопровождать свою продукцию набором программных инструментов, с помощью которых пользователь по определенным правилам и соглашениям мог бы описывать логику работы контроллера. На раннем этапе развития этих программных средств набор поддерживаемых ими функций обеспечивался нестандартными языками. Со временем правила и соглашения совершенствовались и на определенном этапе были оформлены в виде специальных языков программирования, образовав то, что сейчас называется **CASE**-инструментарием.

В 1992 году Международная Электротехническая Комиссия (МЭК, IEC - International Electrotechnical Commission,) взяла под контроль процессы, связанные с развитием этого типа прикладного ПО. Были выдвинуты требования ***открытости системы***, позволяющие унифицировать программные средства и упростить разработку, а именно:

- ✓ возможность разработки драйверов для контроллеров самими пользователями, т.е. сопровождение программных продуктов по программированию контроллеров специальными инструментальными средствами;
- ✓ наличие коммуникационных средств (интерфейсов) для взаимодействия с другими компонентами системы управления;
- ✓ возможность переноса ядра системы на ряд программно-аппаратных платформ.

На рынке появилось большое количество пакетов, удовлетворяющих вышеописанным требованиям. Практически во всех этих пакетах среда разработки реализована в **Windows**-интерфейсе, имеются средства загрузки разработанного приложения в исполнительную систему.

Примеры этих пакетов:

- ✓ RSLogix 500, RS Logix 5, RSLogix 5000 фирмы Rockwell Software для программирования контроллеров различных семейств Allen-Bradley;
- ✓ DirectSOFT для контроллеров семейства Direct Logic фирмы Koyo;
- ✓ пакеты PL7 и Concept - ПО для программирования контроллеров различных семейств компании Schneider Electric;
- ✓ пакеты STEP 5, STEP 7 Micro, STEP 7 для программирования контроллеров семейств S5 и S7 фирмы Siemens;
- ✓ пакет Toolbox для конфигурирования контроллеров семейства Moscad;
- ✓ пакет TelePACE для программирования контроллеров серий TeleSAFE Micro 16 и SCADAPack фирмы Control Microsystems.

Стандартом МЭК 1131-3 определены пять языков программирования контроллеров: три графических (LD, FBD, SFC) и два текстовых (ST, IL).

LD (Ladder Diagram) - графический язык диаграмм релейной логики. Язык LD применяется для описания логических выражений различного уровня сложности. В документации Siemens это редактор контактных планов **LAD**.

FBD (Function Block Diagram) - графический язык функциональных блоковых диаграмм. Язык FBD применяется для построения комплексных процедур, состоящих из различных функциональных библиотечных блоков - арифметических, тригонометрических, регуляторов и т.д.).

SFC (Sequential Function Chart) - графический язык последовательных функциональных схем. Язык SFC предназначен для использования на этапе проектирования ПО и позволяет описать «скелет» программы - логику ее работы на уровне последовательных шагов и условных переходов.

ST (Structured Text) - язык структурированного текста. Это язык высокого уровня, по мнемонике похож на Pascal и применяется для разработки процедур обработки данных.

IL (Instruction List) - язык инструкций. Это язык низкого уровня класса ассемблера и применяется для программирования эффективных, оптимизи-

рованных процедур. В документации Siemens это редактор списка команд **STL**.

В конце 90-х годов появились открытые программные продукты ISaGRAF, InControl (Wonderware), Paradym (Intellution), предназначенные для разработки, отладки и исполнения программ управления как дискретными, так и непрерывными процессами.

В настоящее время все контроллеры и системы управления обслуживаются программными продуктами, реализующими стандарт МЭК 1131-3.

Программные средства верхнего уровня АСУТП (SCADA-пакеты) предназначены для создания прикладного программного обеспечения пультов контроля и управления, реализуемых на различных компьютерных платформах и специализированных рабочих станциях. SCADA - пакеты позволяют при минимальной доле программирования на простых языковых средствах разрабатывать многофункциональный интерфейс, обеспечивающий оператора/диспетчера не только полной информацией о технологическом процессе, но и возможностью им управлять.

В своем развитии SCADA - пакеты прошли тот же путь, что и программное обеспечение для программирования контроллеров. На начальном этапе (80-е годы) фирмы-разработчики аппаратных средств создавали собственные (закрытые) SCADA-системы, способные взаимодействовать только со «своей» аппаратурой. Начиная с 90-х годов, появились универсальные (открытые) SCADA - программы.

Популярные в России SCADA-пакеты.

Trace Mode/Трейс Моуд (AdAstrA) - Россия;

InTouch (Wonderware) - США;

FIX (Intellution) - США;

Genesis (Iconics Co) - США;

Factory Link (United States Data Co) - США;

RealFlex (BJ Software Systems) - США;

Sitex (Jade Software) - Великобритания;

Citect (CI Technology) - Австралия;

WinCC (Siemens) - Германия;

RTWin (SWD Real Time Systems) - Россия;

САРГОН (НВТ - Автоматика) - Россия;

MIK$Sys (МИФИ) - Россия;

Cimplicity (GE Fanuc) - США;

- RSView (Rockwell Automation) - США и многие другие.

SCADA-пакет существует, если с помощью него уже реализовано хотя бы несколько десятков проектов. Вторая предпосылка - нет абсолютно лучшей SCADA-системы для всех случаев применения. SCADA - это всего лишь удобный инструмент в руках разработчика, и ее адаптация к конкретной системе автоматизации - вопрос квалификации и опыта.

Данные технологических процессов специфичны. Они, как правило, могут быть представлены в виде временных рядов «время–значение». Для их сбора и хранения практически любой SCADA-пакет имеет в своем составе подсистему регистрации исторических данных (***архив***) с возможностью последующей выборки требуемых для анализа данных и их представления в виде трендов – графиков или наглядных диаграмм.

Но такие архивы не предназначены для длительного хранения больших объемов информации. К тому же, речь здесь идет о локальных архивах с переменными лишь одного конкретного технологического процесса. Но предприятие имеет в своем составе целый ряд технологических процессов, системы управления которыми выполнены, как правило, на различной программно-аппаратной платформе.

В получении оперативных и объективных технологических данных сегодня заинтересованы практически все службы предприятия. Однако характер необходимой информации различен для различных уровней управления.

На верхнем уровне (***автоматизированная система управления предприятием - АСУП***) нужна только интегрированная (предварительно подготовленная) информация о технологических процессах (данные типа «нарастающим итогом», средних значений за определенные промежутки времени, общее количество произведенных продуктов и т.д.).

Для хранения такой информации хорошо адаптированы базы данных реляционного типа (РБД). Данные в этих базах статичны, связаны многими отношениями, должны быть легко выбираемы по различным сложным критериям. Однако РБД не приспособлены для хранения огромного количества значений параметров, получаемых от SCADA-систем и накапливаемых за достаточно длительное время (до трех и более лет).

В результате, информация, имеющаяся и успешно используемая в АСУТП (АСУ технологическим процессом), недоступна для верхнего уровня

(АСУП).

Таким образом, назрела необходимость создания и внедрения в процесс управления так называемых ***исторических архивов*** производственных данных или ***баз данных реального времени*** (БДРВ) масштаба предприятия. БДРВ должны обеспечить

- ***сбор данных*** с различных источников производственной информации на предприятии (SCADA-систем, DCS-систем, лабораторных систем - LIMS, различных СУБД и т. п.) и их долговременное хранение в едином формате;
- ***доступ к информации*** специалистам и руководителям всех уровней и служб по стандартным протоколам с помощью специализированных клиентских приложений.

Существует целый ряд задач управления, не перекрываемых ни классом АСУП, ни классом АСУТП. Частично эти задачи не перекрываются из-за отсутствия возможностей программного обеспечения этих уровней системы управления. Среди них находятся и задачи, решение которых может оказать решающее влияние на эффективность предприятия в целом:

- диспетчеризация производства,
- оперативное планирование,
- управление качеством продукции и многие другие.

Основные функции SCADA-систем

Программное обеспечение типа **SCADA** предназначено для разработки и эксплуатации автоматизированных систем управления технологическими процессами. SCADA – это новый подход к проблемам человеческого фактора в системах управления (сверху вниз), ориентация в первую очередь на человека (оператора/диспетчера), его задачи и реализуемые им функции.

Такой подход позволил минимизировать участие операторов/диспетчеров в управлении процессом, но оставил за ними право принятия решения в особых ситуациях.

А что дала SCADA-система разработчикам? С появлением SCADA они получили в руки эффективный инструмент для проектирования систем управления, к преимуществам которого можно отнести:

- высокую степень автоматизации процесса разработки системы
- управления;

- ✓ участие в разработке специалистов в области автоматизируемых
- ✓ процессов (программирование без программирования);
- ✓ реальное сокращение временных, а, следовательно, и финансовых
- ✓ затрат на разработку систем управления.

Основные функциональные обязанности самих операторов/диспетчеров:

- ✓ регистрация значений основных технологических и хозрасчетных
  - ○параметров;
- ✓ анализ полученных данных и их сопоставление со сменно-суточными
  - ○заданиями и календарными планами;
- ✓ учет и регистрация причин нарушений хода технологического
  - ○процесса;
- ✓ ведение журналов, составление оперативных рапортов, отчетов
  - ○и других документов;
- ✓ предоставление данных о ходе технологического процесса и
  - ○состоянии оборудования в вышестоящие службы и т. д.

История развития диспетческих служб промышленных предприятий

Раньше в операторной (диспетчерской) находился щит управления (отсюда название – «щитовая»). Для установок и технологических процессов с несколькими сотнями параметров контроля и регулирования длина щита могла достигать нескольких десятков метров, а количество приборов на них измерялось многими десятками, а иногда и сотнями. Среди этих приборов были и показывающие (шкала и указатель), и самопишущие (кроме шкалы и указателя еще и диаграммная бумага с пером), и сигнализирующие. В определенное время оператор, обходя щит, записывал показания приборов в журнал. Так решалась задача **сбора и регистрации** информации.

В приборах, обслуживающих регулируемые параметры, имелись устройства для настройки задания регулятору и для перехода с автоматического режима управления на ручное (дистанционное). Здесь же, рядом с приборами, находились многочисленные кнопки, тумблеры и рубильники для включения и отключения различного технологического оборудования. Таким образом решались задачи **дистанционного управления** технологическими параметрами и оборудованием.

Над щитом управления (как правило, на стене) находилась *мнемосхема* технологического процесса с изображенными на ней технологическими аппаратами, материальными потоками и многочисленными лампами сигнализации зеленого, желтого и красного (аварийного) цвета. Эти лампы начинали

мигать при возникновении нештатной ситуации. В особо опасных ситуациях предусматривалась возможность подачи звукового сигнала (сирена) для быстрого предупреждения всего оперативного персонала. Так решались задачи, связанные с **сигнализацией** нарушений технологического регламента (отклонений текущих значений технологических параметров от заданных, отказа оборудования).

С появлением в операторной/диспетчерской компьютеров было естественным часть функций, связанных со сбором, регистрацией, обработкой и отображением информации, определением нештатных (аварийных) ситуаций, ведением документации, отчетов, переложить на компьютеры. Еще во времена первых управляющих вычислительных машин с монохромными алфавитно-цифровыми дисплеями на этих дисплеях усилиями энтузиастов-разработчиков уже создавались «псевдографические» изображения - прообраз современной графики. Уже тогда системы обеспечивали сбор, обработку, отображение информации, ввод команд и данных оператором, архивирование и протоколирование хода процесса.

Хотелось бы отметить, что с появлением современных программно-технических средств автоматизации, рабочих станций операторов/диспетчеров, функционирующих на базе программного обеспечения SCADA, щиты управления и настенные мнемосхемы не канули безвозвратно в лету. Там, где это продиктовано целесообразностью, щиты и пульты управления остаются, но становятся более компактными.

Появление управляющих вычислительных машин, а затем и персональных компьютеров вовлекло в процесс создания операторского интерфейса программистов. Они хорошо владеют компьютером, языками программирования и способны писать сложные программы. Для этого программисту нужен лишь алгоритм (формализованная схема решения задачи). Но беда в том, что программист, как правило, не владеет технологией, не «понимает» технологического процесса. Поэтому для разработки алгоритмов надо было привлекать специалистов-технологов, например, инженеров по автоматизации.

Выход из этой ситуации был найден в создании методов «программирования без реального программирования», доступных для понимания не только программисту, но и инженеру-технологу. В результате появились программные пакеты для создания интерфейса «человек-машина» (Man/Humain Machine Interface, MMI/HMI). За рубежом это программное обеспечение по-

лучило название SCADA (Supervisory Control And Data Acquisition – супервизорное/диспетчерское управление и сбор данных), так как предназначалось для разработки и функциональной поддержки автоматизированных рабочих мест (АРМов) операторов/диспетчеров в АСУТП. А в середине 90-х аббревиатура SCADA (СКАДА) уверенно появилась и в лексиконе российских специалистов по автоматизации.

Оказалось, что большинство задач, стоящих перед создателями программного обеспечения верхнего уровня АСУ ТП различных отраслей промышленности, достаточно легко *поддается унификации*, потому что функции оператора/диспетчера практически любого производства достаточно унифицированы и легко поддаются формализации.

Таким образом, **базовый набор функций SCADA-систем** предопределен ролью этого программного обеспечения в системах управления (HMI) и реализован практически во всех пакетах. Это:

- ✓ сбор информации с устройств нижнего уровня (датчиков, контроллеров);
- ✓ прием и передача команд оператора/диспетчера на контроллеры и исполнительные устройства (дистанционное управление объектами);
- ✓ сетевое взаимодействие с информационной системой предприятия (с вышестоящими службами);
- ✓ отображение параметров технологического процесса и состояния оборудования с помощью мнемосхем, таблиц, графиков и т.п. в удобной для восприятия форме;
- ✓ оповещение эксплуатационного персонала об аварийных ситуациях и событиях, связанных с контролируемым технологическим процессом и функционированием программно-аппаратных средств АСУ ТП с регистрацией действий персонала в аварийных ситуациях.
- ✓ хранение полученной информации в архивах;
- ✓ представление текущих и накопленных (архивных) данных в виде графиков (тренды);
- ✓ вторичная обработка информации;
- ✓ формирование сводок и других отчетных документов по созданным на этапе проектирования шаблонам.

К интерфейсу, созданному на базе программного обеспечения SCADA, предъявляется несколько фундаментальных требований:

- ✓ он должен быть интуитивно понятен и удобен для оператора/диспетчера;
- ✓ единичная ошибка оператора не должна вызывать выдачу ложной команды управления на объект.

Открытость SCADA-систем

На начальном этапе развития (80-е годы) каждый производитель микропроцессорных систем управления разрабатывал свою собственную SCADA-программу. Такие программы могли взаимодействовать только с узким кругом контроллеров, и по всем параметрам были **закрытыми** (отсутствие набора драйверов для работы с устройствами различных производителей и средств их создания, отсутствие стандартных механизмов взаимодействия с другими программными продуктами и т. д.).

С появлением ***концепции открытых систем*** (начало 90-х) программные средства для операторских станций становятся самостоятельным продуктом.

Одной из первых задач, поставленных перед разработчиками SCADA, стала задача организации многопользовательских систем управления, то есть систем, способных поддерживать достаточно большое количество АРМ пользователей (клиентов). В результате появилась ***клиент - серверная*** технология или архитектура.

Клиент - серверная архитектура (рис. 2.1) характеризуется наличием двух взаимодействующих самостоятельных процессов - клиента и сервера, которые, в общем случае, могут выполняться на разных компьютерах, обмениваясь данными по сети. По такой схеме могут быть построены системы управления технологическими процессами, системы обработки данных на основе систем управления базами данных (СУБД) и т. п.

Клиент-серверная архитектура предполагает, что вся информация о технологическом процессе от контроллеров собирается и обрабатывается на сервере ввода/вывода (сервер базы данных), к которому по сети подключаются АРМ клиентов.

Под ***станцией-сервером*** в этой архитектуре следует понимать компьютер со специальным программным обеспечением для сбора и хранения данных и последующей их передачи по каналам связи оперативному персоналу для контроля и управления технологическим процессом, а также всем заинтересованным специалистам и руководителям.

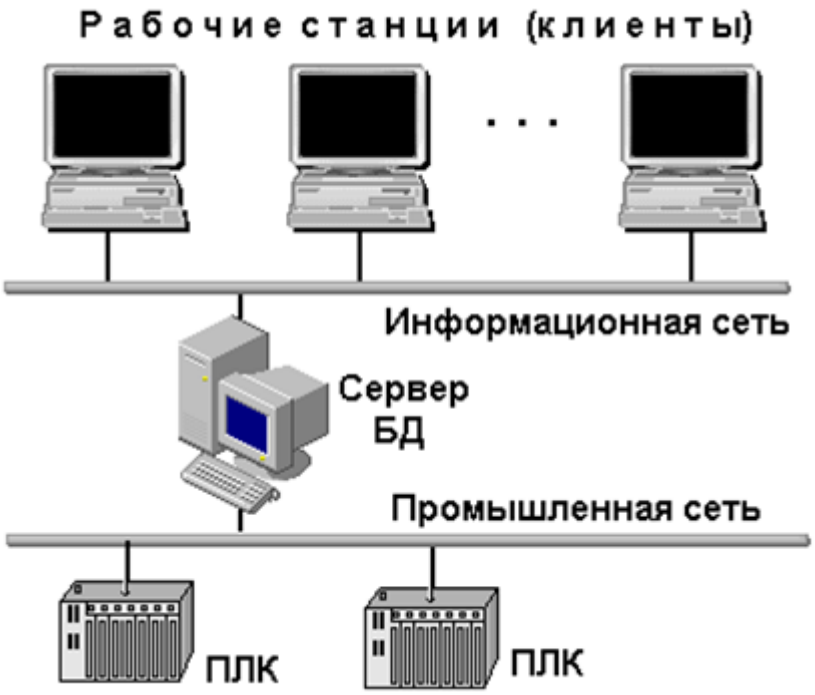

Рис. 2.1. Клиент-серверная архитектура

.

По определению ***сервер*** является ***поставщиком*** информации, а ***клиент*** - ее ***потребителем***. Таким образом, рабочие станции операторов/диспетчеров, специалистов, руководителей являются ***станциями-клиентами***.

Обычно клиентом служит настольный ПК, выполняющий программное обеспечение конечного пользователя. ПО клиента - это любая прикладная программа или пакет, способные направлять запросы по сети серверу и обрабатывать получаемую в ответ информацию. Естественно, функции клиентских станций, а, следовательно, и ***программное обеспечение***, различны и определяются функциями рабочего места, которое они обеспечивают.

Количество операторских станций, серверов ввода/вывода (серверов БД) определяется на стадии проектирования и зависит, прежде всего, от объема перерабатываемой в системе информации. Для небольших систем управления функции сервера ввода/вывода и станции оператора (HMI) могут быть совмещены на одном компьютере.

В сетевых распределенных системах средствами SCADA/HMI стало возможным создавать станции (узлы) различного функционального назначения:

- ✓ станции операторов/диспетчеров,
- ✓ серверы с функциями HMI,

- ✓ “слепые” серверы (без функций HMI),
- ✓ станции мониторинга (только просмотр без прав на управление) для специалистов и руководителей и другие.

Состав SCADA

SCADA-программы имеют в своем составе взаимозависимые модули:

- ✓ **Development** - среда разработки проекта;
- ✓ **Runtime** - среда исполнения;
- ✓ **База данных реального времени.**

В целях снижения стоимости проекта эти модули могут устанавливаться на разные компьютеры. Например, станции оператора, как правило, являются узлами Runtime (или View) с полным набором функций человеко-машинного интерфейса. При этом хотя бы один компьютер в сети должен быть типа Development. На таких узлах проект разрабатывается, корректируется, а также может и исполняться. Некоторые SCADA-системы допускают внесение изменений в проект без остановки работы всей системы. Программное обеспечение SCADA-серверов позволяет создавать полный проект системы управления, включая базу данных и HMI.

Важным аспектом в структурном построении сетевых систем управления является ***структура базы данных реального времени*** *(централизованная или распределенная). Каждая из структур в SCADA/HMI-системах реализуется разными разработчиками по-разному. От реализации существенно зависят эффективность обеспечения единства и целостности базы данных, ее надежность, возможности модификации и т.д.*

В одних случаях для доступа к данным на компьютере-клиенте создается «своя» база данных, копируемая с удаленных серверов. Дублирование данных может привести к определенным проблемам с точки зрения целостности базы данных и производительности системы управления. При модификации базы данных с такой организацией, например, при введении дополнительной переменной потребуются изменения в каждой сетевой копии, использующей эту переменную.

В других случаях компьютерам-клиентам не требуются копии баз данных. Они получают необходимую им информацию по сети от сервера, в задачу которого входит подержание базы данных. Серверов может быть несколько, и любая часть данных хранится только в одном месте, на одном сервере. Поэтому и модификация базы данных производится только на одном компьютере – сервере базы данных, что обеспечивает ее единство и целостность. Такой подход к структурному построению системы снижает нагрузку на сеть и дает еще целый ряд преимуществ.

С точки зрения структурного построения SCADA-пакетов различают:

- ✓ *системы, обеспечивающие полный набор базовых функций HMI* - могут комплектоваться дополнительными опциями, реализующими необязательные в применении функции контроля и управления.
- ✓ *системы, состоящие из модулей, реализующих отдельные функции HMI* – полностью модульная система, состоящая из сервера ввода/вывода, сервера алармов, сервера трендов, и т.д.(рис. 2.2) Для небольших проектов все модули могут исполняться на одном компьютере. В проектах с большим количеством переменных модули можно распределить на несколько компьютеров в разных сочетаниях. Вариант клиент-серверной архитектуры такой системы представлен на рис. 2.2.

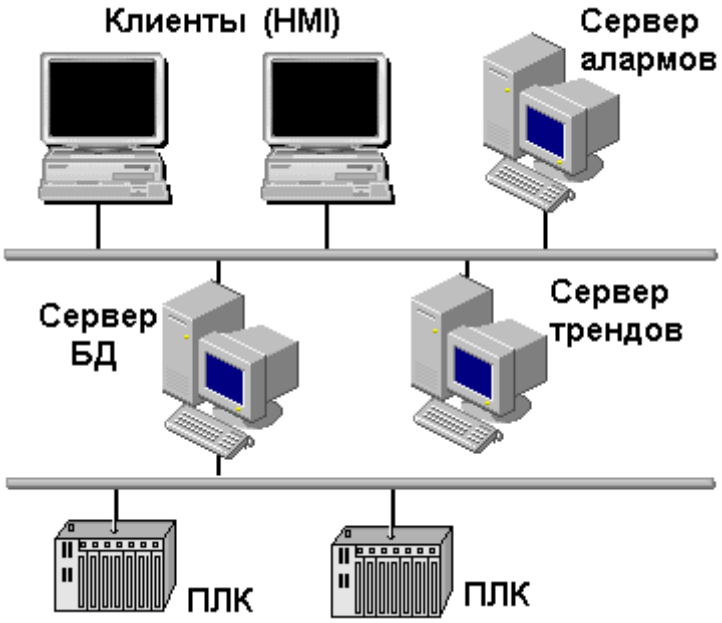

Рис. 2.2. Архитектура модульной SCADA.

В клиент-серверной архитектуре системы управления, представленной на рис. 2.2, функции сбора и хранения данных, управления алармами (тревожными сигналами) и трендами распределены между тремя серверами. Функция HMI реализуется на станциях-клиентах.

Например, SCADA **Citect** имеет в своем составе пять функциональных модулей (серверов или клиентов):

- ✓ **I/O** - сервер ввода/вывода. Обеспечивает передачу данных между физическими устройствами ввода/вывода и другими модулями **Citect.**
- ✓ **Display** - клиент визуализации. Обеспечивает операторский интерфейс:
  - отображение данных, поступающих от других модулей **Citect**, и
  - управление выполнением команд оператора.
- ✓ **Alarms** - сервер алармов –

- отслеживает данные,
- сравнивает их с допустимыми пределами,
- проверяет выполнение заданных условий и
- отображает алармы на соответствующем узле визуализации.

✓ **Trends** - сервер трендов. Собирает и регистрирует трендовую информацию, позволяя отображать развитие процесса в реальном масштабе времени или в ретроспективе.

✓ **Reports** - сервер отчетов. Генерирует отчеты

- по истечении определенного времени,
- при возникновении определенного события или
- по запросу оператора.

В одной сети можно использовать только один сервер алармов, сервер трендов и сервер отчетов. В то же время допускается использование нескольких серверов ввода/вывода (**I/O Server**). Количество компьютеров с установленным модулем **Display** (обеспечивающим операторский интерфейс) в сети практически не ограничено.

SCADA как открытая система

Распространение архитектуры «клиент-сервер» стало возможным благодаря развитию и широкому внедрению в практику концепции открытых систем. Главной причиной появления и развития концепции открытых систем явились ***проблемы взаимодействия программно-аппаратных средств*** в локальных компьютерных сетях. Решить эти проблемы можно было только путем международной стандартизации программных и аппаратных интерфейсов.

Концепция открытых систем предполагает свободное взаимодействие программных средств SCADA с программно-техническими средствами разных производителей. Это актуально, так как для современных систем автоматизации характерна высокая степень интеграции большого количества компонент.

В системе автоматизации (рис. 2.3), кроме объекта управления задействован целый комплекс программно-аппаратных средств:

✓ датчики и исполнительные устройства,

✓ контроллеры, серверы баз данных,

✓ рабочие места операторов,

✓ АРМы специалистов и

✓ АРМы руководителей и т. д.

При этом в одной системе могут быть применены технические средства разных производителей.

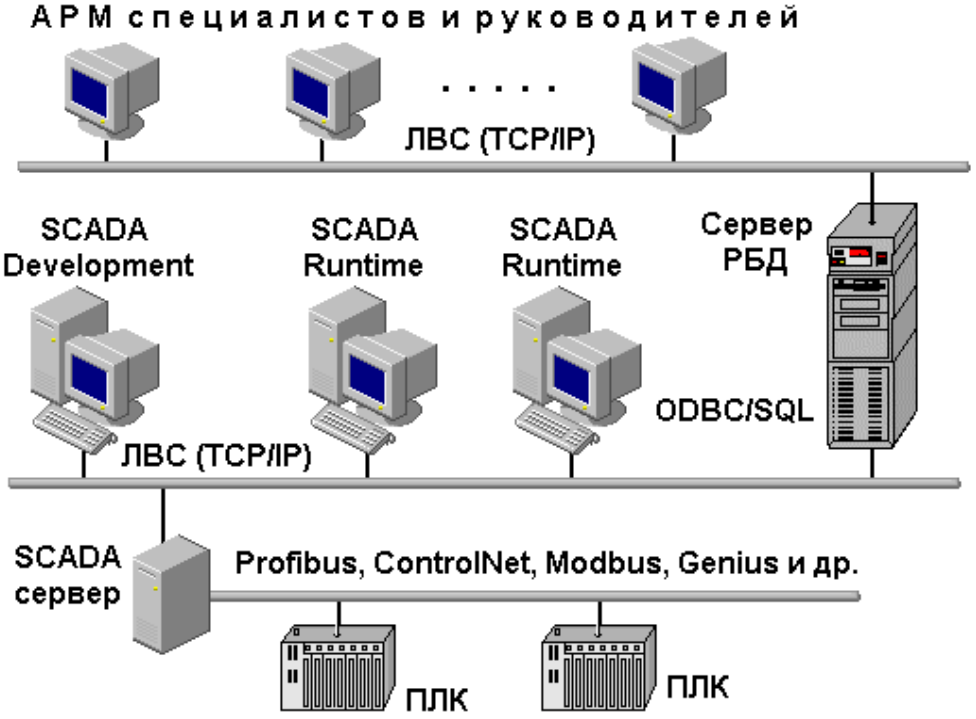

Рис. 2.3. Интеграция SCADA в систему управления.

Для эффективного функционирования в этой разнородной среде SCADA-система должна обеспечивать высокий уровень сетевого взаимодействия. Реализация этой задачи требует от SCADA-системы ***наличия типовых протоколов обмена с наиболее популярными промышленными сетями***, такими, как Profibus, ControlNet, Modbus и другими.

С другой стороны, SCADA-системы должны поддерживать ***интерфейс и со стандартными информационными сетями*** (Ethernet и др.) с использованием стандартных протоколов (TCP/IP и др.) для обмена данными с компонентами распределенной системы управления.

Практически любая SCADA-система имеет в своем составе ***базу данных реального времени*** и ***подсистему архивирования данных***.

Но подсистема архивирования не предназначена для длительного хранения больших массивов информации (месяцы и годы). Информация в ней периодически обновляется, иначе для нее просто не хватит места. SCADA - системы предназначены для обеспечения текущей и архивной информацией оперативного персонала, ответственного за непосредственное управление технологическим процессом.

Информация, отражающая хозяйственную деятельность предприятия (данные для составления материальных балансов установок, производств, предприятия в целом и т. п.), хранится в реляционных базах данных (РБД) типа Oracle, Sybase и т. д. В эти базы данных информация поставляется либо с помощью ручного ввода, либо автоматизированным способом (посредством SCADA-систем). Таким образом, выдвигается еще одно требование к программному обеспечению SCADA - ***наличие в их составе протоколов обмена с типовыми базами данных***.

Наиболее широко применимы два механизма обмена:

- ✓ **ODBC (Open Data Base Connectivity** - взаимодействие с открытыми базами данных) – международный стандарт, предполагающий обмен информацией с РБД посредством ODBC-драйверов. Как стандартный протокол компании Microsoft, ODBC поддерживается и наиболее распространенными приложениями Windows;
- ✓ **SQL (Structured Query Language)** – язык структурированных запросов.

Программное обеспечение SCADA должно взаимодействовать с контроллерами для обеспечения человеко-машинного интерфейса с системой управления (рис. 2.3). К контроллерам через модули ввода/вывода подключены датчики технологических параметров и исполнительные устройства (на рис. 2.3 не показаны).

Информация с датчика записывается в регистр контроллера. Для ее передачи в базу данных SCADA-сервера необходима специальная программа, называемая драйвером. Драйвер, установленный на сервере, обеспечивает обмен данными с контроллером по некоторому физическому каналу. Но для реализации обмена необходим и логический протокол.

После приема SCADA-сервером сигнал попадает в базу данных, где производится его обработка и хранение. Для отображения значения сигнала на мониторе рабочей станции оператора информация с сервера должна быть передана по сети клиентскому компьютеру. И только после этого оператор получит информацию, отображенную изменением значения, цвета, размера, положения и т. п. соответствующего объекта операторского интерфейса.

Большое количество контроллеров с разными программно-аппаратными платформами и постоянное увеличение их числа заставляло разработчиков включать в состав SCADA-системы большое количество готовых драйверов (до нескольких сотен) и инструментарий для разработки собственных драйверов к новым или нестандартным устройствам нижнего уровня.

Для взаимодействия драйверов ввода/вывода и SCADA до недавнего времени использовались два механизма (рис. 2.4):

- ✓ **DDE (Dynamic Data Exchange)** - динамический обмен данными;
- ✓ обмен по собственным (известным только фирме-разработчику) протоколам.

Рис. 2.4. Обмен информацией с помощью DDE-протокола.

Взамен DDE компания Microsoft предложила более эффективное и надежное средство передачи данных между процессами – **OLE** (см. ниже). А вскоре на базе OLE появился новый **стандарт OPC**, ориентированный на рынок промышленной автоматизации.

OPC-интерфейс

OPC – (от **O**LE for **P**rocess **C**ontrol - OLE для управления процессами). Технология OPC основана на разработанной компанией Microsoft технологии **OLE (Object Linking and Embedding**) – встраивание и связывание объектов.

Под объектами здесь подразумеваются так называемые ***компоненты***, которые представляют собой готовые к использованию мини-приложения. Встраивая и связывая эти компоненты, можно разрабатывать **приложения *компонентной* архитектуры**. Этот новый подход к разработке приложений, предложенный компанией Microsoft, получил название технологии **COM (Component Object Model)** – модель компонентных объектов. Теперь приложение-клиент может удаленно вызывать те или иные функции этих объектов так, как будто объекты находятся «рядом». Объект может находиться и в самом деле рядом (в адресном пространстве приложения) - тогда это просто **COM**.

Если же объект находится в другой программе на том же компьютере или на другом узле сети, то это **DCOM-Distributed** (распределенная) **COM**.

Так что же такое **OPC?** OPC представляет собой коммуникационный стандарт, поддерживающий взаимодействие между полевыми устройствами, контроллерами и приложениями разных производителей. Стандарт OPC описывает компонентные объекты, методы и свойства (базирующиеся на технологии OLE/COM) для серверов данных реального времени, таких как PLC,

DCS, систем архивирования данных и других, и обеспечивает передачу информации, содержащейся на этих серверах, стандартным OLE-клиентам.

OPC-взаимодействие основано на клиент-серверной архитектуре.

OPC-клиент (например, **SCADA**), вызывая определенные функции объекта **OPC**-сервера, подписывается на получение определенных данных с определенной частотой. В свою очередь, **OPC**-сервер, опросив физическое устройство, вызывает известные функции клиента, уведомляя его о получении данных и передавая сами данные. Таким образом, при **OPC**-взаимодействии используются как прямые **COM**-вызовы (от клиента к серверу), так и обратные (от сервера к клиенту).

Более популярно изложить идею технологии OPC можно на примере стандартов на шины для персонального компьютера (ПК). К шине ПК можно подключать широкий класс устройств, производимых целым рядом компаний, и все они будут иметь возможность взаимодействовать друг с другом, поскольку используют одну и ту же **стандартную** шину. Также и унифицированный интерфейс OPC позволяет различным программным модулям, производимым самими различными компаниями, взаимодействовать друг с другом.

OPC-сервер отвечает за получение данных от соответствующего устройства управления процессом. На каждом сервере имеется некоторое количество OPC-групп, которые представляют собой логические коллекции данных, запрос на получение которых поступает от клиента. Группы на сервере могут быть доступны нескольким клиентам одновременно или лишь одному клиенту.

Каждая OPC-группа содержит набор OPC-элементов, в которых хранятся данные, поступившие от соответствующего устройства управления процессами. Запрос клиента серверу на получение данных реализуется посредством указания идентификатора элемента. Идентификаторы элементов – свои у каждого сервера. По уникальному идентификатору сервер умеет находить нужное значение в соответствующем устройстве (например, контроллере). Для ПЛК идентификатор элемента обычно соответствует номеру регистра. Дополнительно сервер может снабжать полученные данные меткой времени.

Любое устройство, для которого есть **OPC**-сервер, может использоваться вместе с любой современной **SCADA**-системой, реализованной на платформе MS Windows.

Развивающая стандарт OPC некоммерческая организация OPC Foundation (**http://www.opcfoundation.org**), насчитывает свыше 200 членов. В нее входят почти все ведущие мировые производители программно-аппаратных средств автоматизации.

Хотя стандарт **OPC** и основан на универсальном фундаменте - **COM/DCOM**, он разрабатывался специально для использования в промышленной автоматизации.

Стандарт состоит из трех основных спецификаций:

- ✓ доступ к данным реального времени (Data Access);
- ✓ обработка тревог и событий (Alarms & Events);
- ✓ доступ к историческим данным (Historical Data Access).

Соответственно, OPC-серверов тоже может быть три вида, хотя допускается совмещать все эти функции в одном сервере. OPC-серверы физических устройств обычно являются только серверами данных.

Раньше разработчикам клиентских приложений приходилось писать множество драйверов (верхняя часть рис. 2.5) для взаимодействия с каждым из используемых управляющих устройств (контроллеров).

Стандарт OPC позволяет написать лишь один-единственный драйвер (нижняя часть рис. 2.5) для доступа к данным, поступающим в едином формате от самых различных источников.

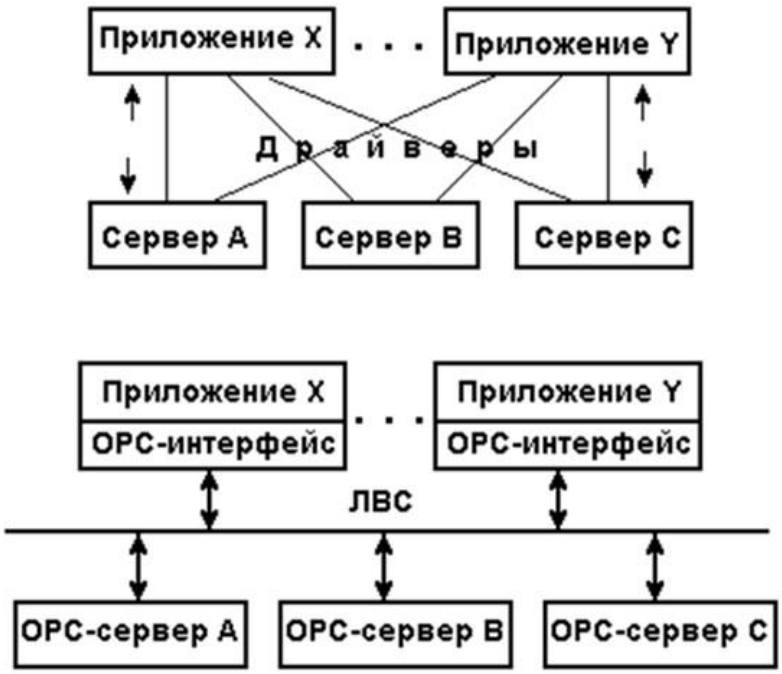

Рис. 2.5. Обмен данными по OPC-интерфейсу.

OPC-интерфейс допускает различные варианты обмена: с физическими устройствами, с распределенными сетевыми системами управления и с любыми приложениями (рис.2.6). На рынке имеются и инструментальные пакеты для написания OPC-компонентов.

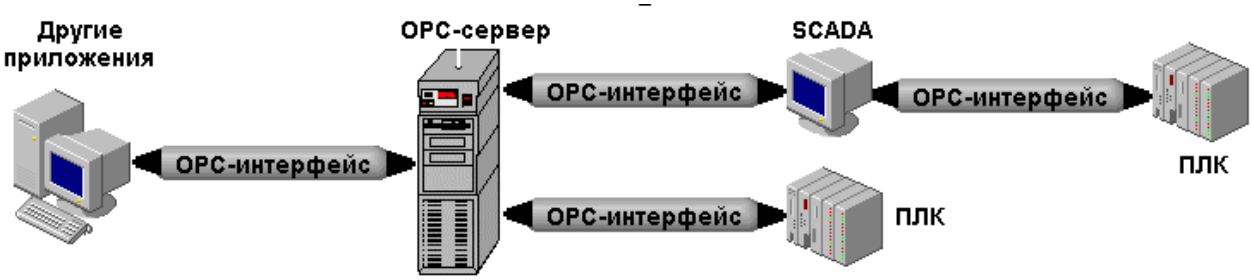

Рис. 2.6. Обмен данными по OPC-интерфейсу.

Использование технологии OPC позволяет конечным пользователям выбирать программно-аппаратные средства, наиболее отвечающие их потребностям, независимо от того, кто их производит.

Другим преимуществом описываемой технологии является то, что при ее использовании снижаются риски и стоимость интеграционных работ. Все используемые в системе компоненты работают на одной и той же технологии.

При разработке систем автоматизации может потребоваться создание собственных программных модулей (не предусмотренных в SCADA- системе) и их включение в систему автоматизации. Поэтому свойство открытости SCADA-систем является очень важной характеристикой программных продуктов этого класса. Открытость SCADA-системы означает возможность доступа к спецификациям системных вызовов, реализующих тот или иной системный сервис. Это может быть и доступ к графическим функциям, функциям работы с базами данных и т.д.

С другой стороны, сегодня в мире существует множество компаний, занимающихся разработкой различных программных компонентов для SCADA-систем, например, ActiveX-объектов. Их использование при разработке систем автоматизации упрощает и ускоряет процесс проектирования. Этот процесс все больше начинает напоминать процесс «сборки» прикладного программного обеспечения из готовых компонентов. Снижаются требования к квалификации программистов – количество задач, решаемых системой с помощью программ собственной разработки на языках высокого уров-

ня типа C или Visual Basic, уменьшается. Все это способствует расширению области применения SCADA-систем.

ActiveX-объекты

ActiveX – это технология Microsoft, основанная на COM/DCOM (см. выше) и предназначенная для написания сетевых приложений. Она предоставляет программистам наборы стандартных библиотек, значительно облегчающих процесс кодирования.

Стандарт ActiveX позволяет программным компонентам взаимодействовать друг с другом по сети независимо от языка программирования, на котором они написаны (Visual Basic, Visual C++, Borland Delphi, Borland C++, любые средства разработки на Java).

ActiveX обеспечивает некий «скрепляющий раствор», с помощью которого отдельные программные компоненты на разных компьютерах «склеиваются» в единую распределенную систему.

Технология ActiveX включает в себя клиентскую и серверную части.

Серверная часть технологии ActiveX реализована с помощью Microsoft **Internet Information Server** (IIS).

SCADA являются контейнерами для ActiveX-объектов. А это значит, что огромное количество готовых к многократному использованию ActiveX-объектов, создаваемых многочисленными производителями подобного программного продукта, могут встраиваться с минимальным программированием в SCADA-приложения. И тогда процесс разработки человеко-машинного интерфейса будет напоминать работу с конструктором, заключающуюся в подборе и встраивании готовых компонентов.

В режиме исполнения ActiveX-компоненты поддерживают динамический обмен данными с другими сетевыми программно-аппаратными компонентами по OPC-интерфейсу.

Пример ActiveX-объекта приведен на рис. 2.7.

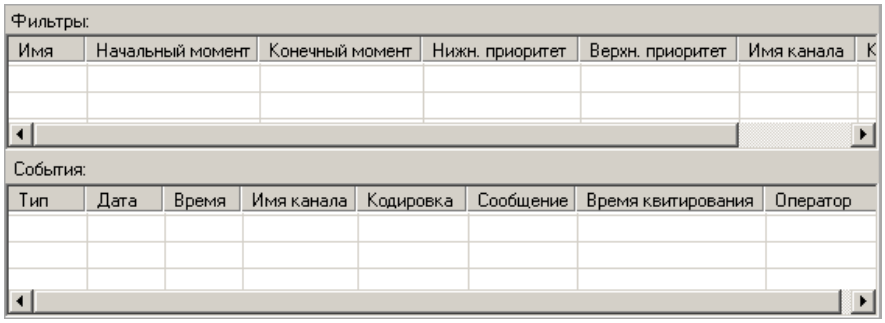

Рис. 2.7. ActiveX-объект «Сводка сигнализации».

Итак, открытость программного обеспечения SCADA обеспечивается целым рядом факторов, а именно:

- ✓ возможностью создания собственных программных модулей и использования программных модулей разработки других компаний;
- ✓ наличием специальных драйверов для связи SCADA с наиболее популярными контроллерами разных фирм;
- ✓ наличием специальных инструментальных средств для создания новых драйверов;
- ✓ возможностью их работы в типовых операционных системах;
- ✓ наличием типовых программных интерфейсов (DDE, OLE, OPC, ActiveX, ODBC, SQL и др.), связывающих ПО SCADA с другими программно-аппаратными средствами системы управления, включая и СУБД.

Сейчас уже можно сказать, что современные системы SCADA/HMI хорошо структурированы и представляют собой готовые к применению и согласованные по функциям и по всем интерфейсам наборы программных продуктов и вспомогательных компонентов.

Организация доступа к SCADA-приложениям

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 (прежде всего, специалистами и руководителями среднего звена).

Для автоматизированного доступа к информации реального времени с любого рабочего места необходимо установить компьютер, подключенный к локальной сети. Организованное таким образом автоматизированное рабочее место (АРМ) предназначено для реализации вполне определенных функций. Поэтому программное обеспечение компьютера (системное и прикладное) должно обеспечить соответствующий данному АРМ набор пользовательских услуг. К их числу можно отнести:

- ✓ объем предоставляемой информации;
- ✓ форма представления информации;
- ✓ реализуемые функции (только информационные или с возможностью выдачи управляющих воздействий);
- ✓ протяженность и надежность канала связи «источник-потребитель»;
- ✓ простота освоения пользователем и т.д.

Системное и прикладное ПО, необходимое компьютеру АРМ для получения удаленного доступа к производственной информации, называется клиентским приложением. 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:

1) с возможностью передачи управляющих воздействий с клиентского приложения и
2) чисто мониторинговые приложения.

Пользователю необходимо лишь определить достаточный набор услуг. Весьма существенным критерием при организации клиентского узла (АРМ) 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».

Самыми простыми и распространенными клиентскими приложениями в настоящее время являются клиенты в локальной сети (рис. 2.8). Такие клиентские приложения в SCADA-системах традиционно объединяются с серверными приложениями протоколами локальных сетей. Часто таким протоколом является TCP/IP.

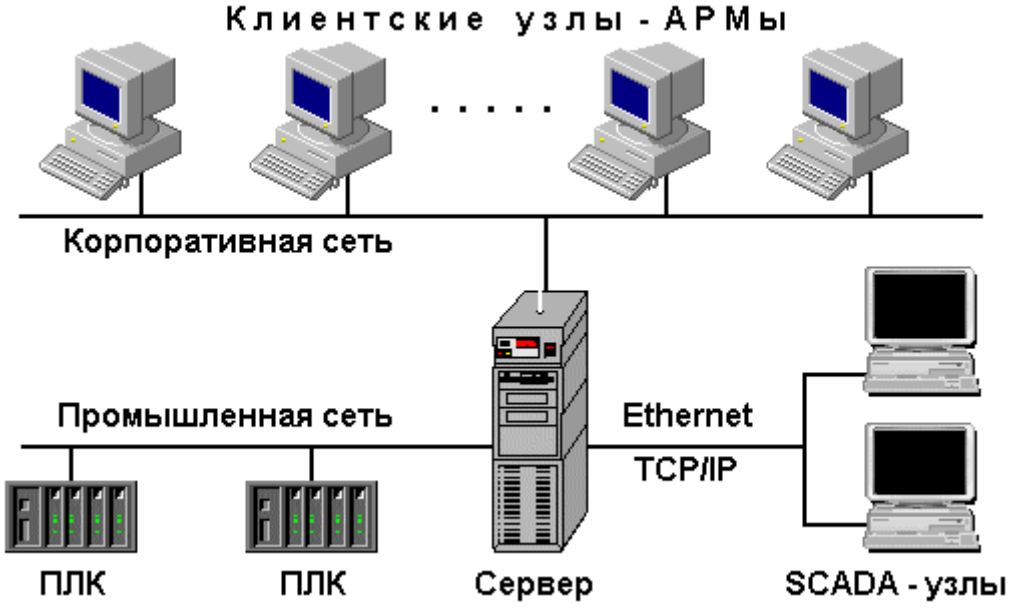

Рис. 2.8. Организация доступа к информации через локальную сеть.

Когда речь идет об организации большого количества автоматизированных рабочих мест на базе программного обеспечения SCADA, то такое решение может оказаться дорогостоящим («богатые» клиенты). К тому же, большинство пользователей SCADA-приложений, в отличие от операторов/диспетчеров, относится к категории нерегулярных, т. е. подключается к системе периодически по мере необходимости.

Технология сервер/терминал

Постоянное появление новых версий программного обеспечения, предъявляющих все более высокие требования к производительности клиентских ПК, привело к тому, что некоторые компании-разработчики программного обеспечения решили разработать технологию, которая бы обеспечила выполнение всех высокопроизводительных вычислений на сервере, оставляя клиентским компьютерам роль терминалов.

Технология сервер/терминал поддерживает режим ***клиентских сессий***, когда один сервер обслуживает несколько клиентов, функционирующих независимо друг от друга. При этом каждый терминал 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. Операции ввода, активизируемые клиентом с клавиатуры и мыши, обслуживаются сервером. Добавление нового клиента заключается лишь в подключении нового терминала к сети.

Терминальные пользователи имеют доступ к данным, мнемосхемам, трендам, алармам с возможностью обмена информацией в реальном времени без необходимости установки SCADA-системы на локальном компьютере (терминале). Таким образом, речь идет о технологиях терминального доступа с использованием так называемых ***«тонких» клиентов***.

Терминал может играть роль как станции оператора/диспетчера, так и АРМ нерегулярных пользователей (технологов, специалистов службы КИП и т. п.), которые могут иметь доступ к необходимой оперативной информации о технологическом процессе и оборудовании (рис. 2.9).

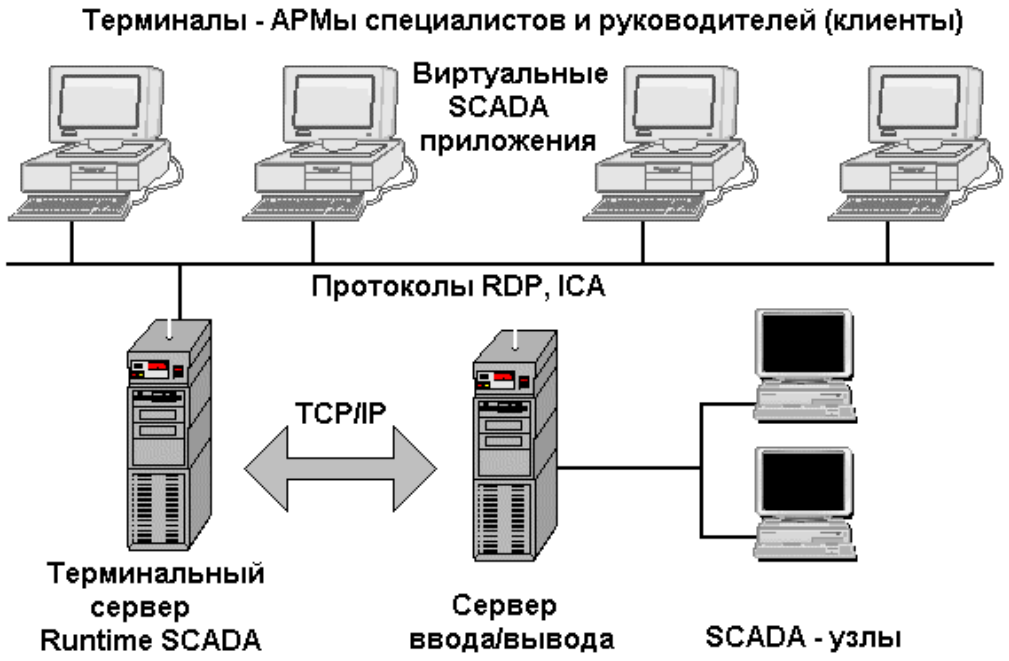

Рис. 2.9. Архитектура терминал-сервер.

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

- приложения устанавливаются и поддерживаются администратором только на сервере;
- обновление программного обеспечения выполняется только один раз и только на сервере;
- терминальные клиенты могут быть реализованы на различных и, что особенно важно, недорогих платформах.

При работе в терминальном режиме вся обработка информации производится на сервере. Его конфигурация зависит от установленных на сервере приложений и от количества обслуживаемых им терминалов. При обработке высокоскоростных приложений для большого количества терминалов (десятки) может потребоваться достаточно дорогостоящий сервер (большая оперативная память).

Internet/Intranet- технологии

Очевидным плюсом сети **Internet** является ее уникальная протяженность и распределенность, что позволяет передавать информацию через тысячи километров между любыми двумя точками земного шара. Кроме этого, сеть отличается уникальной стандартизацией передаваемых данных, что обеспечивает одинаковую читаемость, информативность и однозначность передаваемых данных вне зависимости от операционной системы, в которой

работает компьютер. Эту возможность дает применение стандартного протокола передачи TCP/IP.

Однако наряду с достоинствами **Internet** следует отметить и основной недостаток - очень низкая скорость передачи данных. Сочетание различных физических сред передачи информации и таких свойств протокола TCP/IP как неопределенность времени получения ответа ведут к тому, что передаваемая информация будет передана правильно и без потерь, но заранее сказать, какое время это займет, нельзя. Очевидно, что Internet -технологии мало подойдут для применения в системах с быстротекущими процессами, однако там, где время не является критичным, Internet является приемлемым решением по обеспечению своевременной и точной информацией оператора системы, инженера-технолога или руководителя.

Удобство и популярность Internet стали основной причиной того, что Web-технологии начали активно применяться во внутренних информационных системах предприятий. Каждое предприятие рано или поздно сталкивается с необходимостью автоматизации своей деятельности. Одной из первых ставится задача централизованного хранения информации и доступа к ней. Если раньше такие технологии использовались лишь на самом верхнем уровне управления - АСУП, то в последнее время все большее распространение они получают и в системах уровня АСУ ТП (в системах класса SCADA/HMI).

Внутренние информационные системы предприятия, построенные с использованием Web-технологий, получили собственное название – «Intranet» (интранет - внутренняя сеть). Интранет совсем не обязательно должна ограничиваться локальной сетью предприятия - она может объединять несколько предприятий, находящихся на значительных расстояниях. Отличие Intranet от Internet заключается в том, что ее информационные ресурсы и пользователи объединены общими задачами и принадлежностью одному коллективу.

Самым простым, но очень действенным методом интеграции HMI/SCADA в Интернет является **использование электронной почты** в качестве средства оповещения при появлении новых записей в журнале тревог. Этими возможностями обладают большинство SCADA-систем, имеющихся сейчас на рынке. Электронная почта, кроме прямой посылки письма адресату через Интернет, может использовать и различные «перевалочные пункты», например, шлюзы пейджинговых компаний для посылки сообщения непосредственно на пейджер адресата.

Гораздо более информативной является возможность **генерирования отчетов** о текущем положении дел на объекте в стандарте HTML. Для использования этого метода SCADA-система формирует отчет с диаграммами, графиками, таблицами в виде HTML-файла, который сохраняется на диске (локального или удаленного компьютера). Периодичность обновления отчета зависит только от настроек SCADA-системы и не очень влияет на производительность остальных компонентов системы управления. Сохраненный файл, в свою очередь, может использоваться Web-сервером для предоставления доступа к этим данным через сеть Интернет из любой точки земного шара, используя обыкновенный Web-браузер. Метод не предполагает возможности воздействовать на объект через систему автоматизации, доступны лишь функции мониторинга.

Большие возможности предоставляет **супервизорное управление** через Интернет. Для осуществления этого метода управления системой АСУ ТП необходима SCADA-система, поддерживающая функции управления по сети TCP/IP. При этом функционирующая на удаленном компьютере SCADA- система должна иметь в своем распоряжении копию проекта, включая описание используемых переменных, графические объекты, скрипты и т. п. («толстые» клиенты). В этом случае пересылаемые по сети Internet данные будут содержать только текущие значения параметров, считанных из контроллеров (сбор данных), и команды удаленного компьютера (управление). Примерами реализации таких систем могут служить программы WebCast (фирма Intellution, пакет iFix), NetLink (AdAstra, Trace Mode) и Scout (Wonderware, InTouch).

Другую концепцию предлагает метод связи через **браузер (Web-browser)**. В этом случае используется технология так называемого «тонкого» клиента. При установке связи между Web-браузером и SCADA-сервером в локальный компьютер осуществляется загрузка данных о работающем в системе проекте (включая графические объекты). В этом случае вся математическая обработка данных происходит на удаленном сервере, на локальном же компьютере идет только представление данных, используя ActiveX или другую Web-технологию. Примером реализации могут служить наборы подключаемых модулей WebClient (US Data, FactoryLink/MonitorPro), WebActivator (AdAstra, Trace Mode).

Особое место в Web-технологиях занимает **сбор данных через Интернет от удаленных контроллеров**. Этот метод фактически соответствует

традиционно принятой структуре построения АСУ ТП с использованием SCADA-систем, но в данном случае между самой системой и ПЛК может лежать не одна тысяча километров. В такой конфигурации может работать любая SCADA-система, умеющая посылать сообщения по протоколу TCP/IP (что могут делать практически все системы). Аналогично и ПЛК могут работать в такой системе, если они имеют Ethernet или последовательный порт с поддержкой TCP/IP. Практически все крупнейшие производители контроллеров имеют такие модели.

Совершенно новой технологией для управления через Интернет являются **встраиваемые в ПЛК Web-серверы**. Сейчас можно говорить лишь о наметившихся перспективах. Одна из главных особенностей этой «революционной» технологии (кроме универсальности связи с ПЛК) - отказ от использования SCADA-систем. Web-сервер находится в контроллере, который подключен непосредственно к сети Internet. Имеющийся в контроллере сопроцессор осуществляет формирование необходимых HTML-страниц и связывает их с данными, поступающими с объекта. Однако в данном случае основная тяжесть работы по обработке данных будет ложиться на плечи самого контроллера, который вынужден будет кроме первичной обработки данных осуществлять и вторичную обработку, что может потребовать применения гораздо более мощного процессора ПЛК, чем в случае работы без Web-сервера.

Во всех Internet/Intranet-решениях по обмену данными кроме технологического сервера как ***поставщика*** данных и клиента как ***получателя*** информации задействован ***Web-сервер*** (рис. 2.10). 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.

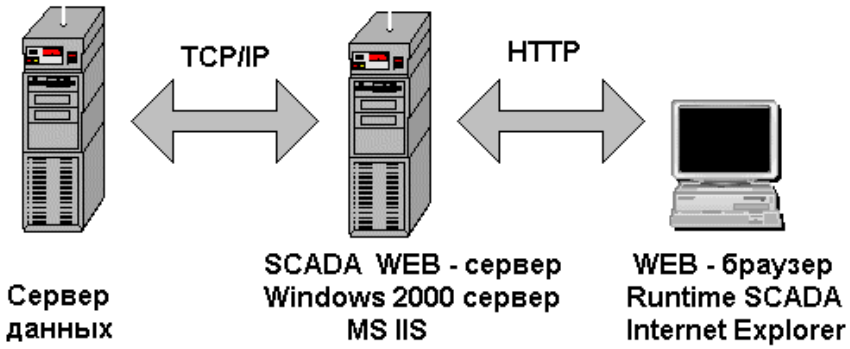

Рис. 2.9. Интеграция SCADA и Internet.

Взаимодействие между Web-сервером и клиентами осуществляется на основе протокола HTTP (HyperText Transfer Protocol - протокол передачи гипертекста).

Интегрированные SCADA-системы

Одним из наиболее значимых факторов развития SCADA-систем становится то, что некоторые ведущие производители ***расширяют функции SCADA*** «***по вертикали***» ***иерархии*** многоуровневой системы управления.

С одной стороны, идет расширение функций в сторону непосредственного управления технологическими процессами (автоматическое регулирование и программно-логическое управление). Функции непосредственного управления реализуются в пакетах прикладных программ как для контроллеров, построенных на основе PC-совместимых контроллеров (SoftPLC), так и для компьютерной реализации функций непосредственного управления (SoftControl).

Широкое использование IBM PC платформы в контроллерах (softlogic) началось в 90-х годах XX века и было обусловлено многими факторами, один из которых – лучшее соотношение «производительность - цена». А для России того времени это было определяющим. И вот отечественная фирма AdAstrA интегрирует свою SCADA-систему с системой программирования PC-контроллеров. Так появилась новая технология сквозного программирования компонентов нижнего и верхнего уровней АСУТП.

Говоря о компьютерной реализации функций непосредственного автоматического управления технологическим процессом, следует отметить, что практически все известные инструментальные SCADA-системы обеспечивают эту возможность. Хотя такое совмещение и позволяет экономить на аппаратных средствах, оно может иметь ряд негативных последствий. Во-первых, операционная система операторской станции может не обеспечить необходимую для конкретного технологического процесса скорость реакции SCADA-системы. Во-вторых, никто не гарантирован от «зависания» системы, хотя для некоторых объектов (инерционных) это может быть и не критично.

С другой стороны, первичная информация, собранная SCADA-системами от технологических установок и производств для принятия оперативных (тактических) решений на уровне операторов/диспетчеров, должна быть доступна в **реальном времени всем уровням управления** с целью ее анализа и принятия управленческих (стратегических) решений.

Для ее решения на рынке программных продуктов стал появляться новый класс программного обеспечения – интегрированные пакеты промышленной автоматизации. В этих пакетах SCADA/HMI является лишь одним из компонентов. Другой важнейший компонент таких систем – базы данных реального времени или архивы исторических данных, предназначенные для хранения огромных массивов информации с возможностью доступа к ней с различных АРМ. Сюда же можно отнести специализированные пакеты для управления периодическими процессами, для выявления и минимизации простоев оборудования, для просмотра производственной информации с помощью Интернет-технологий и т. п.

К классу интегрированных систем можно отнести такие программные продукты ведущих производителей SCADA, как FIX Dynamics (Intellution/GE Fanuc), FactorySuite 2000 (Wonderware) и другие. Эти системы представляют собой мощные программные комплексы, обеспечивающие интеграцию системы управления производством в целом. Использование в системах разных уровней единого стиля оформления, единой терминологии, инструментария, служебных средств и т. д. значительно облегчают разработчикам проектирование систем, а предприятиям - их освоение и эксплуатацию.

Надежность SCADA-систем

Понятие ***надежности*** SCADA-пакетов включает в себя два аспекта:

- ✓ надежность самого программного продукта SCADA – определяется
  - надежностью операционной системы,
  - наличием средств сохранения данных и конфигурации при сбоях,
  - наличием средств автоматического перезапуска системы
- ✓ возможность программного резервирования компонентов системы в различных вариантах.

По надежности современные SCADA-продукты, также как и по функциональности, незначительно отличаются друг от друга. Тем не менее, при выборе пакета можно обратить внимание на список его внедрений. Наличие в таком списке проектов для опасных и ответственных производств, проектов с большим числом параметров, территориально и функционально распределенных АРМов говорит о достаточно высокой надежности SCADA-пакета.

Но система управления может полностью выйти из строя не только по причине отказа программного обеспечения, но и оборудования.

Получившая наиболее широкое распространение распределенная система управления, представленная на рис. 2.11, выйдет из строя, если всего лишь в одном компоненте (сервере) возникнет неисправность.

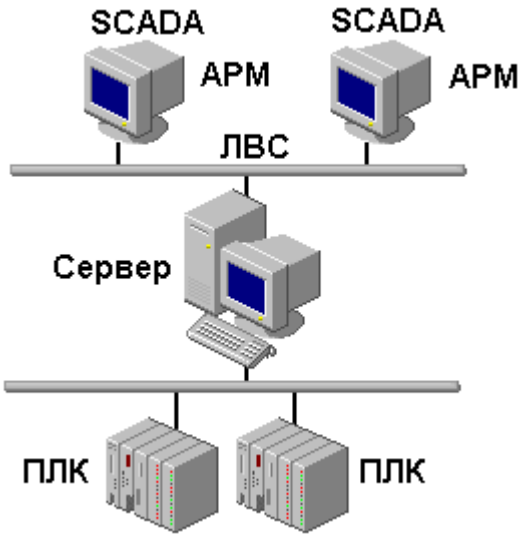

Рис. 2.11. Сетевая архитектура SCADA

Реализация SCADA-пакетами функций резервирования позволяет устранять отказы в системе без потери ее функциональных возможностей и производительности. Программное обеспечение SCADA поддерживает реализацию резервирования различных компонентов системы управления как вследствие особенности архитектуры, так и наличия встроенных механизмов.

Дублирование сервера ввода/вывода

Для повышения надежности системы управления достаточно явно просматривается вариант с резервированием сервера (рис. 2.12). Здесь возможны два варианта. В одном случае оба сервера (основной и резервный) взаимодействуют с устройствами ввода/вывода, удваивая нагрузку на промышленную сеть и снижая производительность системы. В штатном режиме клиенты взаимодействуют с основным сервером. При выходе его из строя они направляют свои запросы к резервному серверу.

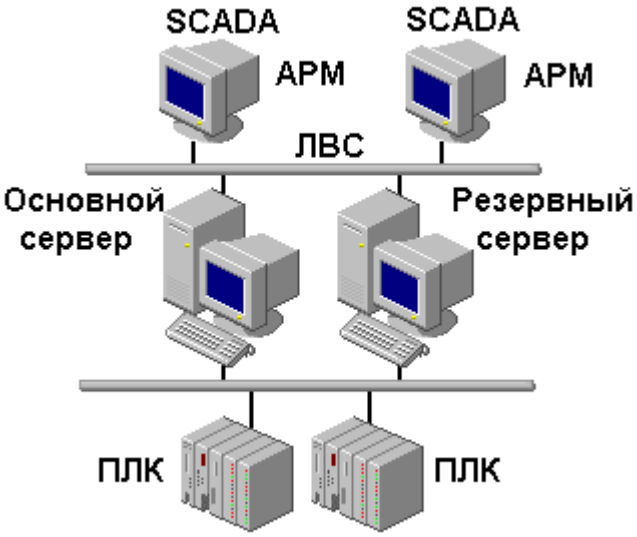

Рис. 2.11. Резервирование сервера.

В распределенной клиент-серверной архитектуре SCADA-систем лишь один (основной) сервер взаимодействует с контроллерами. При этом основной сервер постоянно обновляет базу данных резервного сервера, обеспечивая его постоянную готовность.

Резервирование сети и контроллеров

Структура, приведенная на рис. 2.11, увеличивает надежность системы, устраняя одно из основных «слабых» мест – отказ сервера. Другим «слабым» местом распределенной системы управления может быть сама сеть. Выход ее из строя нарушает управление, так как станции операторов/диспетчеров в этом случае оказываются отрезанными от системы. Повышение надежности системы управления обеспечивается дополнительной сетью (рис. 2.12).

Большинство контроллеров может поддерживать дополнительную (резервную) связь с сервером ввода/вывода. При отказе основного канала гарантируется обмен данными между контроллером и сервером Достичь полного резервирования можно путем дублирования контроллеров (рис. 2.13).

. Рассмотренные выше способы повышения надежности системы управления хорошо известны. Важным здесь является то, что именно встроенные в SCADA-систему механизмы позволяют конфигурировать распределенную клиент-серверную архитектуру, определяя на стадии проектирования основные и резервные устройства системы управления.

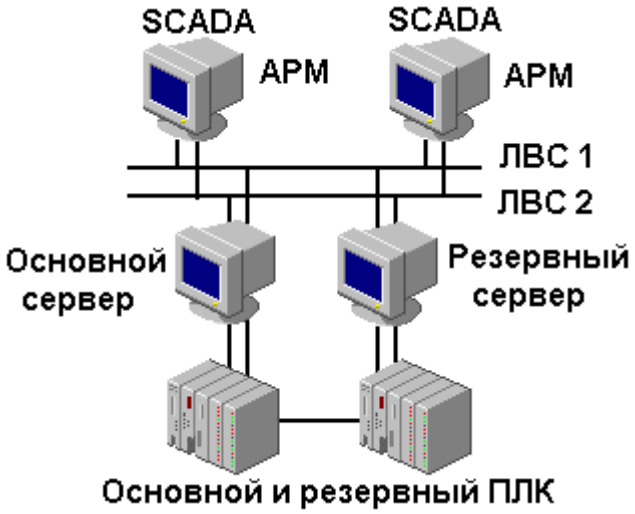

Рис. 2.13. Варианты резервирования.

А в режиме исполнения именно SCADA- система определяет неисправность того или иного компонента системы и автоматически производит переключение на резервное оборудование, предупреждая об этом оперативный персонал.

Программно-аппаратная платформа

К этой группе можно отнести следующие характеристики: компьютерная платформа, операционная система, конфигурация компьютера (частота процессора, требуемые ресурсы оперативной и дисковой памяти), возможность переноса приложений в другую операционную систему.

Анализ платформ и операционных систем необходим, поскольку они определяют возможность распространения SCADA-системы на имеющиеся вычислительные средства и стоимость системы.

Программное обеспечение SCADA, как и любое другое ПО, выполняется под управлением той или иной операционной системы. Какая же операционная система наиболее приемлема для программного обеспечения верхнего уровня? Обязательно применение ОСРВ или достаточно операционной системы общего назначения? Этот вопрос обсуждался на протяжении нескольких лет в различных периодических изданиях, посвященных автоматизации технологических процессов. В итоге, компромисс найден: требования к параметрам операционной системы должны определяться автоматизируемым объектом и прикладной задачей.

Нельзя также и забывать, что неотъемлемой частью верхнего уровня АСУ ТП является человек, время реакции которого на события недетермини-

ровано и зачастую достаточно велико. И, наконец, нельзя не учитывать тенденции развития мирового рынка программного обеспечения.

В результате, подавляющее большинство SCADA-систем реализовано на MS Windows-платформах (Windows NT/2000). Это и InTouch, и FIX, и Genesis, и российский Трейс Моуд. Из четырнадцати систем, приведенных выше, двенадцать предназначены для работы в различных вариантах ОС MS Windows. Здесь, безусловно, сказались позиции компании Microsoft на рынке операционных систем. Известно, что именно компания Microsoft была и остается «законодателем моды» в этом классе программного обеспечения.

Компьютерные ресурсы, требуемые для установки и нормального функционирования различных компонентов SCADA-систем, определяются многими факторами, в том числе, назначением сетевого компьютера (рабочая станция оператора, сервер БД, АРМ специалиста и т. п.), количеством обрабатываемых переменных, используемой операционной системой (Windows 95/98/NT/2000, QNX) и т. п.

Могут накладываться также ограничения на качество и объем памяти видеокарты, разрешение экрана монитора, размеры монитора.

Требования к аппаратным средствам, призванным поддерживать серверные функции, могут быть существенно более высокими. Это относится и к объему оперативной памяти, и к объему жесткого диска, который может измеряться уже десятками и сотнями Гб. Требования к компьютеру для WinCC V6.0:

Рекомендуется

- ✓ Pentium
- ✓ Основная память минимум 128 MB, рекомендуется 256 MB
- ✓ Винчестер >1 GB
- ✓ Дисковод CD ROM
- ✓ Свободное пространство на жестком диске 170 MB

С другой стороны, многие клиентские компьютеры при использовании современных сетевых технологий, таких, как архитектура Server/Terminal, Internet-технологий (WEB-сервер), могут быть достаточно слабых конфигураций (IBM 286/386) с минимальными требованиями как к оперативной, так и к дисковой памяти, а то и вовсе бездисковыми.

Масштабируемость - это способность ПО SCADA наращивать размеры системы управления, обеспечивая при этом преемственность по отношению ко всем ранее установленным программно-аппаратным средствам.

С ростом мощности компьютеров и соответствующим ростом информационной мощности операторских станций SCADA-системы становятся масштабируемыми. Они выпускаются в различных вариантах, которые при сохранении в целом функционального профиля поддерживают от нескольких десятков или сотен до десятков тысяч входов/выходов (лицензируемых точек).

Естественно, стоимость таких пакетов различна: чем больше переменных поддерживает SCADA-пакет, тем он дороже. Но это удобно потребителю - можно приобрести пакет под проект практически любого масштаба.

Градация количества лицензируемых точек в различных SCADA-пакетах различна. В ряде пакетов она более равномерна, чем в других. Например, на рынке программных продуктов можно найти SCADA-пакеты на 75, 150, 500, 1 500, 5 000, 15 000, 50 000, 150 000 и 450 000 переменных. При этом учитываются только внешние переменные, считываемые с устройств ввода/вывода. Внутренние переменные, которые будут определены разработчиком при проектировании, не являются лицензируемыми (бесплатны), хотя и будут храниться в памяти компьютера или на жестком диске. Другие фирмы-производители SCADA в общее количество лицензируемых точек включают и внутренние переменные. Например, приобретение такого пакета на 500 лицензируемых точек означает следующее. Если в соответствии с проектом разработчику потребуется создать 100 внутренних переменных, то система способна будет обрабатывать лишь 400 переменных ввода/вывода. Но и о возможном расширении системы не надо забывать.

При расширении системы управления, например, увеличении количества обрабатываемых переменных, создании новых станций для перераспределения вычислительной нагрузки между компьютерами в системе SCADA-пакеты снабжаются встроенными механизмами, которые позволяют разработчикам реализовать такие возможности. С точки зрения удобства использования этих механизмов все SCADA-пакеты различны. Многие фирмы предлагают системы, в которых основная работа по конфигурированию компьютеров клиент-серверной архитектуры хорошо автоматизирована.

Эксплуатационные характеристики

К этой группе можно отнести:

- ✓ ***удобство интерфейса*** среды разработки (это качество обеспечивается применением Windows –подобных интерфейсов), полнота и наглядность представления функций системы на экране, удобство и информативность контекстных и оперативных подсказок, справочной системы;
- ✓ ***качество документации*** - полнота, ясность и наглядность описания системы, применение установившейся терминологии, русификация, уровень русификации (экраны, подсказки, справочная система, системные сообщения, документация);
- ✓ ***полнота/недостаточность средств диагностики*** состояния системы при сбоях и отказах, нарушениях внешних связей;
- ✓ трудоемкость и уровень автоматизации работ при инсталляции и конфигурировании системы; возможности внесения изменений в систему без ее остановки и т.д.
- ✓ положение программного продукта на рынке: дилерская сеть, консультационная поддержка, наличие «горячей линии», обучение, условия обновления версий (upgrade), количество инсталляций и т. д.

Эксплуатационные характеристики в значительной мере носят субъективный характер и не могут быть оценены количественно. О них можно судить только по результатам практического использования программного продукта: тестирования, апробирования, анализа, опыта промышленного внедрения. Косвенной характеристикой качества и отработанности крупнотиражного программного продукта служит его положение на рынке, поскольку большое число реализаций продукта свидетельствует о солидном опыте применений, учтенном при обновлениях продукта.

Основные подсистемы SCADA-пакетов

Создание современной системы управления потребует от разработчика некоторого набора знаний применяемого в проекте SCADA-пакета. Что же надо знать о SCADA разработчику, приступая к созданию проекта?

Для реализации базовых функций SCADA-системы разработчику потребуется, как минимум:

- ✓ организовать взаимодействие SCADA-пакета с аппаратными средствами автоматизации (контроллерами);
- ✓ создать графический интерфейс для диспетчера/оператора, т.е. отображе-

ние технологического процесса и значений параметров на динамизированных мнемосхемах;

- ✓ обеспечить оперативный персонал информацией о ситуациях, связанных с отклонением технологических параметров от заданных значений, о предаварийном состоянии оборудования и т.п.;
- ✓ настроить систему регистрации и архивирования данных и их представление на мониторе в виде трендов, что позволит оператору и специалистам проводить анализ состояния процесса и оборудования.

Можно перечислить еще ряд типовых задач, решаемых в процессе разработки системы управления (шаблоны отчетов, статистическая обработка данных, взаимодействие с РБД и др.).

Для разработки качественного операторского интерфейса разработчику также необходимо владеть встроенным в SCADA-пакет языком программирования. С его помощью создаются так называемые сценарии (скрипты) – фрагменты программ, обеспечивающие оперативный персонал своевременной информацией и облегчающие управление процессом.

Таким образом, SCADA – это набор инструментов (подсистем) для решения перечисленных выше задач.

Графический интерфейс

Качество отображения информации на мнемосхемах определяется характеристиками графических возможностей пакетов. К ним можно отнести графический редактор, возможность создания объемных изображений, наличие библиотек и разнообразие графических заготовок и готовых объектов, богатство инструментария, многообразие динамических свойств элементов мнемосхем, форматы импортируемых изображений, наличие инструментария для создания растровых рисунков, наличие и возможности многооконных режимов и т. п.

При создании компонентов операторских интерфейсов (например, мнемосхем) разработчику приходится использовать графические объекты, представляющие собой технологические аппараты (колонны, емкости, теплообменники и т. д.), участки трубопровода и такие устройства, как клапаны, насосы, электродвигатели, контроллеры, компьютеры 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 проекта. Для ускорения работы над проек-

том практически все SCADA-пакеты предлагает разработчику библиотеки готовых объектов, включающие сотни и тысячи графических компонентов (рис. 2.14).

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

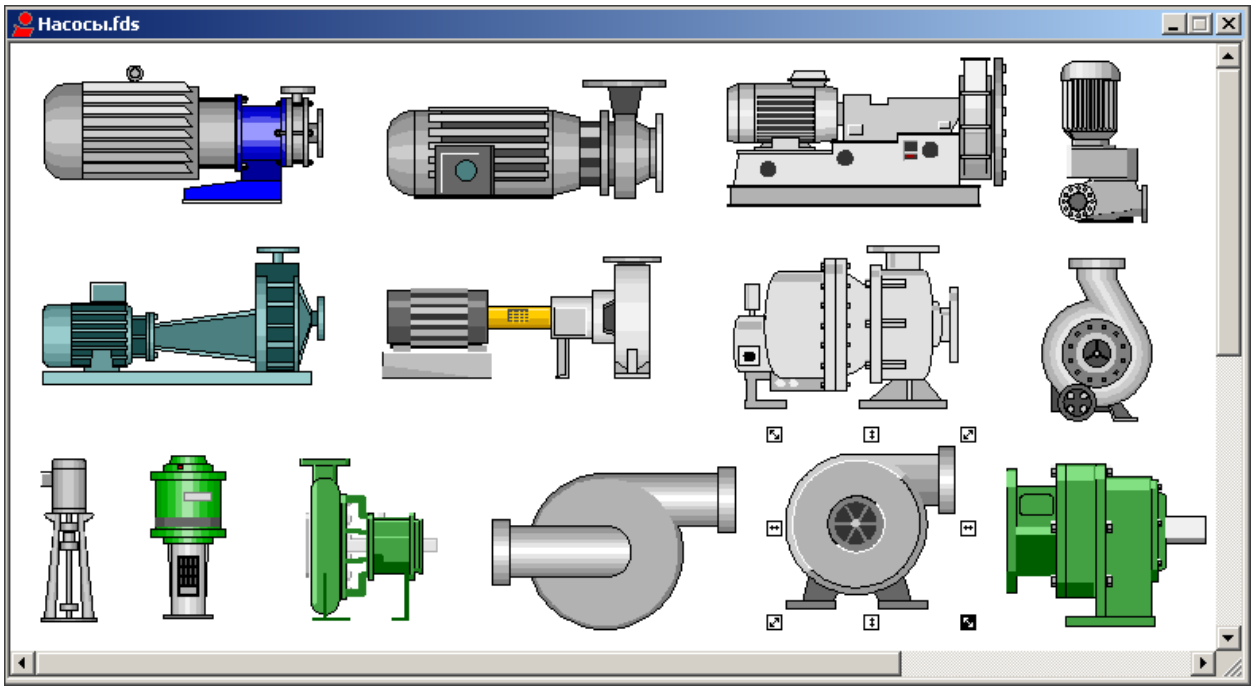

Рис. 2.14. Библиотека «Насосы» SCADA-пакета iFIX.

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 динамические свойства). Но таких объектов в одном окне может оказаться несколько. Время специалиста в этом случае будет расходоваться неэффективно.

Для решения подобных задач SCADA-пакеты предлагают различные

решения:

- ✓ готовые сложные объекты с заданным набором динамических свойств, хранящиеся в специальных библиотеках;
- ✓ инструментарий для их создания с возможностью сохранения в библиотеке для многократного использования.

Разработчику надо лишь выбрать требуемый объект из библиотеки, вставить его в графическую страницу и в появившийся на экране диалог ввести имя/имена переменной/переменных.

В SCADA-системах различных производителей набор динамических свойств объектов достаточно типизирован. В режиме исполнения при определенных условиях объекты интерфейса могут:

- ✓ перемещаться (горизонтально, вертикально);
- ✓ изменять размеры (по горизонтали, по вертикали);
- ✓ заполняться цветом (по горизонтали, по вертикали);
- ✓ быть ползунковыми регуляторами (горизонтального или вертикального типа);
- ✓ появляться на экране и исчезать с него (видимость);
- ✓ мерцать;
- ✓ вращаться;
- ✓ изменять цвет.

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, облегчая оператору/диспетчеру восприятие информации.

В целях унификации окон интерфейса оператора/диспетчера и сокращения сроков разработки проектов некоторые компании-производители SCADA снабжают свои пакеты программ шаблонами окон с возможностью их модификации и создания собственных шаблонов. Другие SCADA-системы предусматривают возможность импорта/экспорта окон из одних приложений в другие, что также существенно упрощает процесс разработки.

Подсистема сигнализации

Возможности по предоставлению информации эксплуатационному персоналу об аварийных ситуациях и событиях обеспечиваются подсистемами сигнализации. Такие подсистемы - обязательный компонент любого SCADA-пакета, но механизмы их реализации различны.

В русском языке понятие «сигнализация» стоит рядом с понятием «тревога». Английским аналогом этих понятий является Alarm (аларм).

Поддерживаемые типы алармов (тревог), приоритеты, возможности по фильтрации алармов (группировка), механизмы вывода информации об алармах, удобство конфигурирования системы алармов и т. п. - вот далеко не полный перечень характеристик подсистемы сигнализации.

Аларм (состояние тревоги) - это сообщение, формируемое системой управления и имеющее целью привлечь внимание оперативного персонала о возникновении ситуации, которая может привести к нарушению технологического процесса или более серьезным последствиям. Степень важности того или иного аварийного сообщения зависит от последствий, к которым может привести нарушение, вызвавшее данное аварийное сообщение. Наиболее важные аварийные сообщения могут потребовать вмешательства оперативного персонала. Поэтому для большинства аварийных сообщений, сформированных системой, требуется подтверждение (квитирование) их получения оператором/диспетчером.

Наряду с алармами в SCADA - системах существует понятие событий. Под с**обытием** следует понимать обычные статусные сообщения системы, не требующие подтверждения их получения и ответной реакции оператора. Обычно события генерируются при возникновении в системе определенных условий (регистрация оператора в системе, ввод информации оператором).

Причины, вызывающие состояние аларма, могут быть самыми разными:

- ✓ отказ аппаратных средств (датчиков, контроллеров, каналов связи);
- ✓ отказ технологического оборудования (насоса, электродвигателя и т. п.);
- ✓ выход параметров технологического процесса за заданные границы.

Все SCADA - системы поддерживают алармы двух типов: **дискретные** и **аналоговые**.

Дискретные алармы срабатывают при изменении состояния дискретной переменной (кран открыт/закрыт, насос включен/выключен). По умолчанию дискретный аларм может срабатывать при переходе на **1** (**ON)** или на **0** (**OFF)**, в зависимости от конкретного SCADA - пакета.

Аналоговые алармы базируются на анализе выхода значений переменной за указанные верхние и нижние пределы. Аналоговые алармы могут быть заданы в нескольких комбинациях (рис. 2.15)**:**

- ✓ верхние пределы (предаварийный и аварийный);
- ✓ нижний пределы (предаварийный и аварийный);

- ✓ отклонение от заданного значения;
- ✓ скорость изменения параметра.

Для выхода переменной из состояния аларма необходимо, чтобы ее значение стало меньше порогового на величину, называемую зоной нечувствительности. Аналогично можно интерпретировать нижние предаварийные и аварийные алармы.

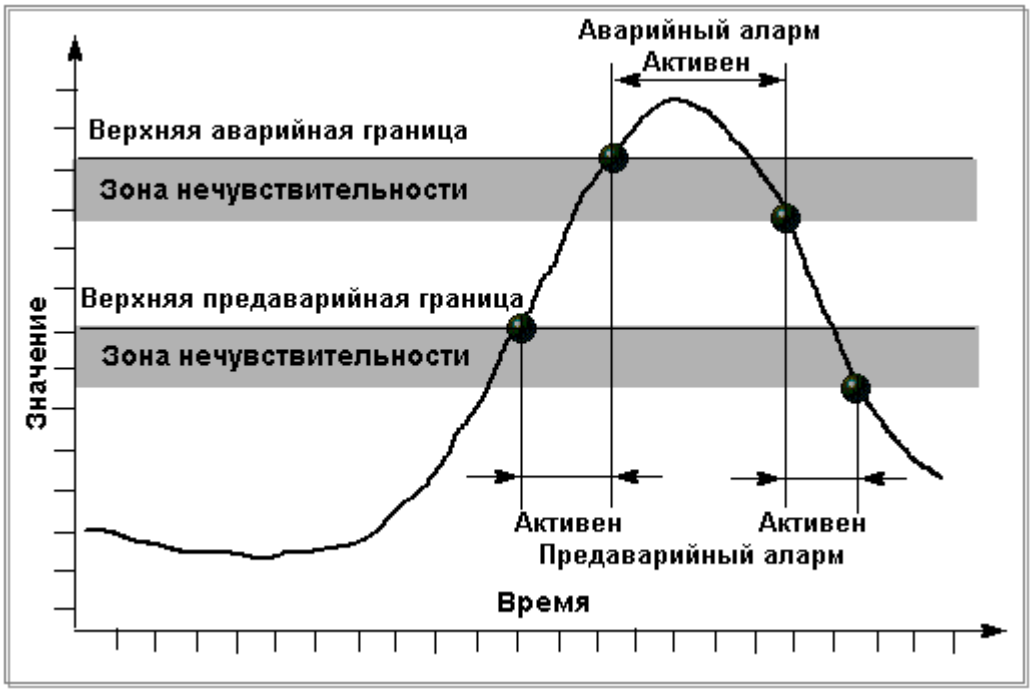

Рис. 2.15. Графическая интерпретация верхних предаварийного и аварийного алармов.

Все вышеизложенное справедливо и для аларма типа «отклонение». **З**аданное значение в ходе технологического процесса может изменяться либо оператором, либо программно (автоматически). Аларм «сработает» при выходе значения переменной за границу допустимого отклонения.

Алармы, определяемые скоростью изменения параметра, возникают в случае, если она становится больше (меньше) предельно допустимой. Понятие «зона нечувствительности» к алармам этого типа не применяется.

Важную роль в подсистеме алармов любого SCADA-пакета играют ***приоритеты***. Приоритеты алармов могут быть использованы в различных целях:

- ✓ для определения способа вывода алармов (на принтер, в файл, в текущую сводку),

- ✓ для определения порядка их появления в окнах текущих алармов,
- ✓ для запуска скриптов,
- ✓ для определения действия, вызываемого срабатыванием аларма определенного приоритета (например, включение звукового сигнала) и т. п.

Как правило, важность приоритета уменьшается с увеличением его значения. Таким образом, приоритет с номером 1 - самый высокий. Например, если алармы с приоритетами от 1 до 10 должны выводиться на экран, то первыми будут выводиться алармы с приоритетом 1 в порядке их поступления, затем - алармы с приоритетом 2 и т. д. Количество значений (уровней) приоритетов в разных SCADA-пакетах различно (десятки и сотни).

Подсистема алармов предусматривает возможность классификации алармов по самым различным признакам: по аппаратам технологического процесса, по типу алармов, имени, приоритету и т. д. В зависимости от этого каждый аларм может быть отнесен к определенной группе (зоне, категории). Подобная **группировка** – удобный способ фильтрации алармов и их обработки (подтверждение, способ вывода, формат, цвет и т. п.).

Вывод информации об аварийных ситуациях реализуется различными способами. Ее можно выводить в специализированные окна операторского интерфейса в виде текущих и архивных сводок, записывать в файлы, распечатывать на принтерах, предназначенных для вывода аварийных сообщений.

Кроме того, эту аварийную информацию можно отображать непосредственно на мнемосхемах интерфейса оперативного персонала:

- ✓ вывод в специальные текстовые поля;
- ✓ динамизация объектов (изменение цвета, мерцание и т .п.).

Формат вывода (информация, включаемая в аварийное сообщение) определяется на стадии проектирования. В строку аварийного сообщения можно включить:

- ✓ текущую время и дату,
- ✓ тип аларма, его приоритет,
- ✓ имя переменной, ее текущее значение, зону нечувствительности, размерность,
- ✓ группу алармов и его состояние (подтвержден/неподтвержден).

Для дискретных алармов можно создать поле **on (**вкл**.)/off** (выкл.). Для алармов с метками времени в поле текущего времени можно выводить информацию с точностью до миллисекунд.

Пример объекта вывода аварийной информации приведен на рис. 2.16.

MM/DD	HH:MM:SS	EVT	Type	Operator	Pri	Name	GroupName	Value/Limit
MM/DD	HH:MM:SS	EVT	Type	Operator	Pri	Name	GroupName	Value/Limit
MM/DD	HH:MM:SS	EVT	Type	Operator	Pri	Name	GroupName	Value/Limit
MM/DD	HH:MM:SS	EVT	Type	Operator	Pri	Name	GroupName	Value/Limit
MM/DD	HH:MM:SS	EVT	Type	Operator	Pri	Name	GroupName	Value/Limit

Рис. 2.16. Пример формата вывода информации об алармах.

Подсистема регистрации, архивирования и отображения данных в виде трендов

Представление данных в виде графиков (трендов) реализуется в современных SCADA-пакетах специальными подсистемами. К характеристикам таких подсистем можно отнести

- ✓ способы регистрации архивных данных,
- ✓ способы отображения трендов,
- ✓ удобство по конфигурированию трендов,
- ✓ возможности по переконфигурированию трендов в режиме Runtime,
- ✓ предоставляемый сервис при работе с архивными трендами,
- ✓ возможность построения графиков y(x) и т. п.

Тренды реального времени **(**Real Tim**e)** отображают динамические изменения параметра в текущем времени (в темпе с протеканием технологического процесса). При появлении нового значения параметра в окне тренда происходит прокрутка графика. Текущее значение параметра выводится, как правило, в правой части окна.

Исторические (Historical**)** ***или архивные тренды*** не являются динамическими и строятся на основе выборки архивных данных. Отображаемые значения переменных на архивных трендах неподвижны и могут быть отображены только на определенном выборкой отрезке времени.

При работе SCADA-системы в режиме Runtime (среда исполнения) производится запись значений переменных в регистрационные файлы.

Для записи значений переменных в регистрационный файл могут использоваться различные способы:

- ✓ регистрация при изменении переменной на величину, превышающую некоторый порог;
- ✓ периодически с заданной частотой.

Предпочтительна регистрация данных в несколько небольших по размеру файлов, чем в один большой файл, т. к. при этом проще осуществлять выборку данных для последующего анализа. Объем выборки для хранения в файлах задается в процессе конфигурирования системы временным периодом (от нескольких часов до недель).

Некоторые SCADA-пакеты используют круговую систему записи в файлы. При этом определяется количество файлов, продолжительность регистрации в каждый из них, время смены регистрационного файла. После того, как истечет время регистрации в последний файл, регистрация будет продолжена снова в первый файл, уничтожая при этом старую информацию (рис. 2.17).

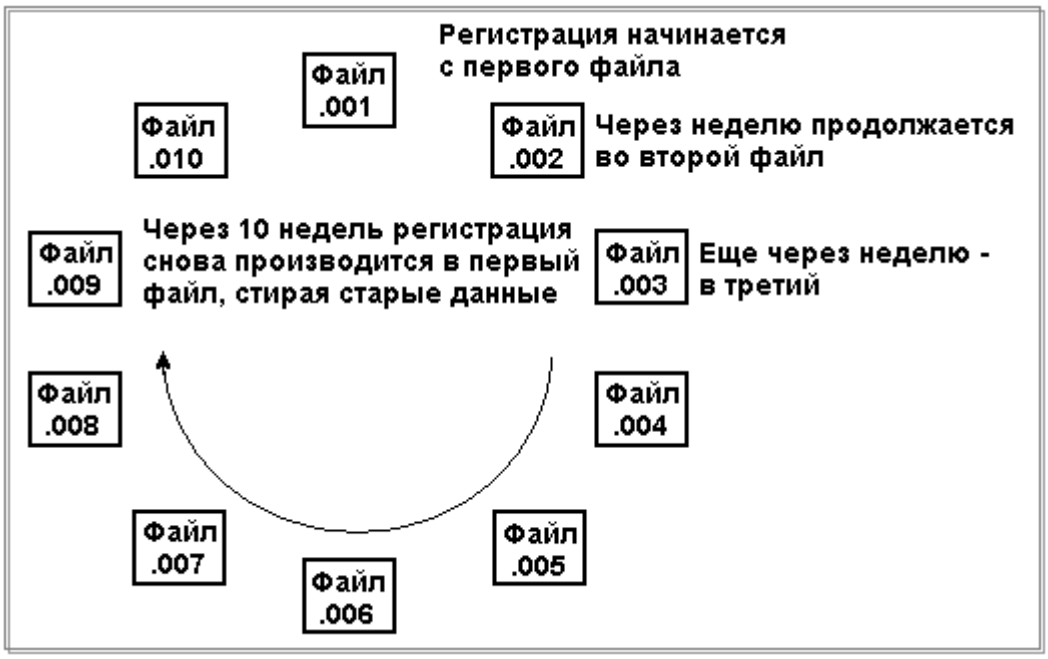

Рис. 2.17. Круговая система регистрации данных.

Разновидностью кругового является метод, когда на стадии проектирования определяется количество файлов, продолжительность регистрации в каждый из них, а также длительность хранения архивных данных для каждого файла. По мере истечения срока хранения информации файлы автоматически уничтожаются, обеспечивая свободное дисковое пространство для вновь создаваемых регистрационных файлов.

Для графического отображения информации SCADA-системы различных производителей предлагают два решения:

- ✓ использование двух различных инструментов для создания диаграмм под тренды реального времени и архивные тренды;
- ✓ единый инструмент для трендов реального времени и архивных трендов.

По числу перьев на одной диаграмме также возможны варианты. В одних SCADA-системах количество перьев на диаграмме задано жестко (4, 8, 16 перьев). Другие предлагают диаграммы на неограниченное количество перьев. Настройка диаграмм производится в специальных диалогах. Параметрами настройки могут быть

- ✓ диапазон времени, охватываемый трендом,
- ✓ частота вывода значений переменной (период обновления),
- ✓ разрешение сетки по горизонтальной и вертикальной осям,
- ✓ параметры перьев: маркеры, стиль и толщина линии, цвет.

Возможность переконфигурирования перьев тренда в режиме Runtime – важная характеристика SCADA-пакета. Она закладывается на стадии проектирования с использованием различных методов: с помощью встроенных функций, уникальных встроенных механизмов.

Удобным механизмом работы с диаграммой в режиме выполнения является отображение курсора времени (визира). В местах пересечения курсора с кривыми высвечиваются значение переменной и время, соответствующее этому значению. Полезной может оказаться и возможность вывода на одной диаграмме перьев с различными пределами отображаемых переменных и различными шкалами.

Для работы с архивными трендами производители SCADA-систем предлагают дополнительный сервис:

- ✓ возможность выделять различные участки диаграммы,
- ✓ увеличивать выделенные участки для детального анализа кривых,
- ✓ перемещаться вдоль архивного тренда и т. п.

Встроенные языки программирования

Встроенные языки программирования - мощное средство SCADA - систем, предоставляющее разработчику гибкий инструмент для разработки сложных приложений. В ранних версиях SCADA - систем языки реализовывали небогатый набор функций. Современные SCADA - системы с точки зрения функциональных возможностей существенно богаче.

Многие функции присутствуют практически во всех языках: математические функции, функции управления экранами, алармами, трендами, ActiveX - объектами, DDE - обменом и т. д.

Полный набор требуемых функций конкретной системы управления обычно не может быть обеспечен только базовым ПО. Существует большое количество задач, в том числе и расчетных, для решения которых потребуется встроенный в SCADA-систему язык программирования. Для повышения функциональности интерфейса в разрабатываемом приложении могут создаваться программные фрагменты (***скрипты***), выполнение которых связывается с разнообразными событиями в приложении, такими как нажатие кнопки, открытие окна и т.п.

Следует отметить, что SCADA, безусловно, упрощает процесс создания ПО верхнего уровня АСУТП. Но этот процесс остается достаточно трудоемким, требующим соответствующей квалификации и времени.

Библиография

1. Андреев Е.Б. Современные технологии автоматизации (5 лекций).
2. Андреев Е.Б. SCADA-системы: взгляд изнутри/Е.Б. Андреев, Н.А. Куцевич, О.В. Синенко. – М.:Издательство «РТСофт», 2004. – 176 с.: ил.