Министерство Общего и Профессионального Образования Российской Федерации

Ивановский Государственный Энергетический Университет

Кафедра Электроники и Микропроцессорных систем

по курсу: «Системы контроля и визуализации» на тему:

«SCADA-системы»

Выполнил студент гр. 1-34M

Шмаргун А.Н.

Проверил:

Анисимов А. А.

Иваново 2003

Содержание:

Введение 2

АСУ ТП и диспетчерское управление 2

Компоненты систем контроля и управления и их назначение 4

Разработка прикладного программного обеспечения СКУ: выбор пути и инструментария 7

Технические характеристики 8

Открытость систем 9

Стоимостные характеристики 10

Эксплуатационные характеристики 10

Графический интерфейс 11

Графические средства InTouch 11

Окна в InTouch 11

Объекты и их свойства 13

Организация взаимодействия с контроллерами 16

Аппаратная реализация связи с устройствами ввода/вывода 17

Серверы ввода/вывода в InTouch 18

Поддерживаемые коммуникационные протоколы 18

Особенности адресации в InTouch 20

Обмен данными с другими приложениями 21

Определение имени доступа в словаре переменных InTouch 22

Тренды в SCADA - системах 25

Тренды в InTouch 25

Архивирование (регистрация) значений переменной 26

Отображение трендов 26

Изменение параметров архивных трендов в режиме исполнения 29

Система распределенных архивов 29

Встроенные языки программирования 30

Скрипты в InTouch 31

Типы скриптов 31

Встроенные функции 32

Функции Quick Functions 36

Разработка графопостроителя в системе InTouch 37

Разработка DDE-сервера 37

Разработка DDE - клиента 39

Список литературы 41

# Введение

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

## АСУ ТП и диспетчерское управление

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

* Первый этап отражает внедрение систем автоматического регулирования (САР). Объектами управления на этом этапе являются отдельные параметры, установки, агрегаты; решение задач стабилизации, программного управления, слежения переходит от человека к САР. У человека появляются функции расчета задания и параметры настройки регуляторов.
* Второй этап - автоматизация технологических процессов. Объектом управления становится рассредоточенная в пространстве система; с помощью систем автоматического управления (САУ) реализуются все более сложные законы управления, решаются задачи оптимального и адаптивного управления, проводится идентификация объекта и состояний системы. Характерной особенностью этого этапа является внедрение систем телемеханики в управление технологическими процессами. Человек все больше отдаляется от объекта управления, между объектом и диспетчером выстраивается целый ряд измерительных систем, исполнительных механизмов, средств телемеханики, мнемосхем и других средств отображения информации (СОИ).
* Третий этап - автоматизированные системы управления технологическими процессами - характеризуется внедрением в управление технологическими процессами вычислительной техники. Вначале - применение микропроцессоров, использование на отдельных фазах управления вычислительных систем; затем активное развитие человеко-машинных систем управления, инженерной психологии, методов и моделей исследования операций и, наконец, диспетчерское управление на основе использования автоматических информационных систем сбора данных и современных вычислительных комплексов.

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

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

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

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

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

Статистика говорит, что за тридцать лет число учтенных аварий удваивается примерно каждые десять лет. В основе любой аварии за исключением стихийных бедствий лежит ошибка человека.

В результате анализа большинства аварий и происшествий на всех видах транспорта, в промышленности и энергетике были получены интересные данные. В 60 - х годах ошибка человека была первоначальной причиной аварий лишь в 20% случаев, тогда как к концу 80-х доля "человеческого фактора" стала приближаться к 80 %.

Одна из причин этой тенденции - старый традиционный подход к построению сложных систем управления, т. е. ориентация на применение новейших технических и технологических достижений и недооценка необходимости построения эффективного человеко - машинного интерфейса, ориентированного на человека (диспетчера).

Таким образом, требование повышения надежности систем диспетчерского управления является одной из предпосылок появления нового подхода при разработке таких систем: ориентация на оператора/диспетчера и его задачи.

Концепция SCАDA (Supervisory Control And Data Acquisition - диспетчерское управление и сбор данных) предопределена всем ходом развития систем управления и результатами научно-технического прогресса. Применение SCADA-технологий позволяет достичь высокого уровня автоматизации в решении задач разработки систем управления, сбора, обработки, передачи, хранения и отображения информации.

Дружественность человеко-машинного интерфейса (HMI/MMI), предоставляемого SCADA - системами, полнота и наглядность представляемой на экране информации, доступность "рычагов" управления, удобство пользования подсказками и справочной системой и т. д. - повышает эффективность взаимодействия диспетчера с системой и сводит к нулю его критические ошибки при управлении.

Следует отметить, что концепция SCADA, основу которой составляет автоматизированная разработка систем управления, позволяет решить еще ряд задач, долгое время считавшихся неразрешимыми: сократить сроки разработки проектов по автоматизации и прямые финансовые затраты на их разработку.

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

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

В России диспетчерское управление технологическими процессами опиралось, главным образом, на опыт оперативно-диспетчерского персонала. Поэтому переход к управлению на основе SCADA-систем стал осуществляться несколько позднее. К трудностям освоения в России новой информационной технологии, какой являются SCADA-системы, относится как отсутствие эксплуатационного опыта, так и недостаток информации о различных SCADA-системах. В мире насчитывается не один десяток компаний, активно занимающихся разработкой и внедрением SCADA-систем. Каждая SCADA-система - это "know-how" компании и поэтому данные о той или иной системе не столь обширны.

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

* выбора SCADA-системы (исходя из требований и особенностей технологического процесса);
* кадрового сопровождения.

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

Подготовка специалистов по разработке и эксплуатации систем управления на базе программного обеспечения SCADA осуществляется на специализированных курсах различных фирм, курсах повышения квалификации. В настоящее время в учебные планы ряда технических университетов начали вводиться дисциплины, связанные с изучением SCADA-систем. Однако специальная литература по SCADA-системам отсутствует; имеются лишь отдельные статьи и рекламные проспекты.

## Компоненты систем контроля и управления и их назначение

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

|  |
| --- |
| Рис.1. Обобщенная схема системы контроля и управления. |

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

* Нижний уровень - уровень объекта (контроллерный) - включает различные датчики для сбора информации о ходе технологического процесса, электроприводы и исполнительные механизмы для реализации регулирующих и управляющих воздействий. Датчики поставляют информацию локальным программируемым логическим контроллерам (PLC - Programming Logical Controoller), которые могут выполнять следующие функции:
  + сбор и обработка информации о параметрах технологического процесса;
  + управление электроприводами и другими исполнительными механизмами;
  + решение задач автоматического логического управления и др.

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

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

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

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

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

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

* Информация с локальных контроллеров может направляться в сеть диспетчерского пункта непосредственно, а также через контроллеры верхнего уровня (см. рис.). В зависимости от поставленной задачи контроллеры верхнего уровня (концентраторы, интеллектуальные или коммуникационные контроллеры) реализуют различные функции. Некоторые из них перечислены ниже:
  + сбор данных с локальных контроллеров;
  + обработка данных, включая масштабирование;
  + поддержание единого времени в системе;
  + синхронизация работы подсистем;
  + организация архивов по выбранным параметрам;
  + обмен информацией между локальными контроллерами и верхним уровнем;
  + работа в автономном режиме при нарушениях связи с верхним уровнем;
  + резервирование каналов передачи данных и др.
* Верхний уровень - диспетчерский пункт (ДП) - включает, прежде всего, одну или несколько станций управления, представляющих собой автоматизированное рабочее место (АРМ) диспетчера/оператора. Здесь же может быть размещен сервер базы данных, рабочие места (компьютеры) для специалистов и т. д. Часто в качестве рабочих станций используются ПЭВМ типа IBM PC различных конфигураций.   
    
  Станции управления предназначены для отображения хода технологического процесса и оперативного управления. Эти задачи и призваны решать SCADA - системы. SCADА - это специализированное программное обеспечение, ориентированное на обеспечение интерфейса между диспетчером и системой управления, а также коммуникацию с внешним миром.

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

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

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

* Все компоненты системы управления объединены между собой каналами связи. Обеспечение взаимодействия SCADA - систем с локальными контроллерами, контроллерами верхнего уровня, офисными и промышленными сетями возложено на так называемое коммуникационное ПО. Это достаточно широкий класс программного обеспечения, выбор которого для конкретной системы управления определяется многими факторами, в том числе и типом применяемых контроллеров, и используемой SCADA - системой. Более подробная информация о коммуникационном ПО приведена в главе 2.
* Большой объем информации, непрерывно поступающий с устройств ввода/вывода систем управления, предопределяет наличие в таких системах баз данных (БД). Основная задача баз данных - своевременно обеспечить пользователя всех уровней управления требуемой информацией. Но если на верхних уровнях АСУ эта задача решена с помощью традиционных БД, то этого не скажешь об уровне АСУ ТП. До недавнего времени регистрация информации в реальном времени решалась на базе ПО интеллектуальных контроллеров и SCADA - систем. В последнее время появились новые возможности по обеспечению высокоскоростного хранения информации в БД. Более подробная информация по базам данных реального времени приведена в главе 6.
* Бурное развитие Интернет не могло не привлечь внимание производителей программного продукта SCADA. Возможно ли применение Интернет - технологий в системах управления технологическими процессами? Если да, то какие решения предлагаются в настоящее время компаниями - разработчиками? Обсуждению этих вопросов посвящена глава 7.

## Разработка прикладного программного обеспечения СКУ: выбор пути и инструментария

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

* Программирование с использованием "традиционных" средств (традиционные языки программирования, стандартные средства отладки и пр.)
* Использование существующих, готовых - COTS (Commercial Of The Shelf) - инструментальных проблемно-ориентированных средств.

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

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

Итак, выбор пути сделан! Это очень важно, но тогда следует сделать и второй шаг - "определиться" с инструментальными средствами разработки ППО.   
Программные продукты класса SCADA широко представлены на мировом рынке. Это несколько десятков SCADA - систем, многие из которых нашли свое применение и в России. Наиболее популярные из них приведены ниже:

* InTouch (Wonderware) - США;
* Citect (CI Technology) - Австралия;
* FIX (Intellution ) - США;
* Genesis (Iconics Co) - США;
* Factory Link (United States Data Co) - США;
* RealFlex (BJ Software Systems) - США;
* Sitex (Jade Software) - Великобритания;
* TraceMode (AdAstrA) - Россия;
* Cimplicity (GE Fanuc) - США;
* САРГОН (НВТ - Автоматика) - Россия.

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

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

* технические характеристики;
* стоимостные характеристики;
* эксплуатационные характеристики.

## Технические характеристики

Программно-аппаратные платформы для SCADA-систем.

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

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

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

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

Имеющиеся средства сетевой поддержки.

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

Встроенные командные языки.

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

Поддерживаемые базы данных.

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

Практически все SCADA-системы, в частности, Genesis, InTouch, Citect, используют ANSI SQL синтаксис, который является независимым от типа базы данных. Таким образом, приложения виртуально изолированы, что позволяет менять базу данных без серьезного изменения самой прикладной задачи, создавать независимые программы для анализа информации, использовать уже наработанное программное обеспечение, ориентированное на обработку данных.

Графические возможности.

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

Крайне важен также вопрос о поддержке в рассматриваемых системах стандартных функций GUI (Graphic Users Interface). Поскольку большинство рассматриваемых SCADA-систем работают под управлением Windows, это и определяет тип используемого GUI.

## Открытость систем

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

Разработка собственных программных модулей.

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

Драйверы ввода-вывода.

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

Разработки третьих фирм.

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

## Стоимостные характеристики

При оценке стоимости SCADA-систем нужно учитывать следующие факторы:

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

## Эксплуатационные характеристики

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

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

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

* Разработка архитектуры системы автоматизации в целом. На этом этапе определяется функциональное назначение каждого узла системы автоматизации.
* Решение вопросов, связанных с возможной поддержкой распределенной архитектуры, необходимостью введения узлов с "горячим резервированием" и т.п.
* Создание прикладной системы управления для каждого узла. На этом этапе специалист в области автоматизируемых процессов наполняет узлы архитектуры алгоритмами, совокупность которых позволяет решать задачи автоматизации.
* Приведение в соответствие параметров прикладной системы с информацией, которой обмениваются устройства нижнего уровня (например, программируемые логические контроллеры - ПЛК) с внешним миром (датчики технологических параметров, исполнительные устройства и др.)
* Отладка созданной прикладной программы в режиме эмуляции. В последующих главах на примере двух известных и хорошо зарекомендовавших себя SCADA-систем (InTouch и Citect) рассмотрены основные компоненты, функции и возможности систем диспетчерского управления и сбора данных.

# Графический интерфейс

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

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

## Графические средства InTouch

Компоненты среды разработки InTouch:

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

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

## Окна в InTouch

Свойства каждого окна (наличие заголовка, цвет фона, размеры и т. д.) определяются при его создании. Создание нового окна производится в среде разработки WindowMaker щелчком по иконке панели инструментов General или командой File/New Window. На экране появится диалог Window Properties (Свойства окна, рис. 2).

|  |
| --- |
| Рис. 2. Диалог Window Properties (Свойства окна). |

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

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

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

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

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

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

По умолчанию при создании нового окна эти параметры примут значения предыдущего (последнего) созданного окна.

Кнопка Scripts (скрипты) дает возможность войти в диалог Window Script для создания оконного сценария.

Для унификации внешнего вида окон приложения и сокращения сроков разработки приложений InTouch предлагает несколько приемов.

Один из таких приемов - дублирование окон. Создание копий окон выполняется командой File/ Save Window As. Для быстрого доступа к этой команде можно воспользоваться меню правой кнопки мыши (см. ниже).

Второй прием, который также позволяет экономить время разработки приложения - импорт окон. Можно повторно использовать все ранее созданные окна, объекты и скрипты. Чтобы импортировать окна из другого InTouch - приложения, необходимо воспользоваться командой File/ Import.

Интерфейс WindowMaker с открытым окном представлен на рис. 3.

|  |
| --- |
| Рис. 3. Интерфейс WindowMaker. |

Сверху экрана расположена строка меню, включающая опции для работы с окнами, редактирования и выравнивания объектов в окне, настройки инструментариев, текста, толщины и стиля линий и т. д.

Слева от рабочего поля видно меню Application Explorer, которое может быть выведено в интерфейс WindowMaker или закрыто нажатием соответствующей иконки инструментария.

## Объекты и их свойства

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

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

Текст на кнопке редактируется с помощью команды Special/Substitute Strings... При этом текстовое поле может содержать только одну строку.   
  
Один и тот же объект может иметь набор различных динамических свойств. Комбинации этих свойств предоставляют возможность создавать на экране в режиме исполнения (Runtime) практически любые динамические эффекты. Для установки динамических свойств надо прежде всего вызвать на экран диалог их выбора (рис.4). Это достигается командой Special/Animation Link или двойным щелчком левой кнопки мыши на объекте.

|  |
| --- |
| Рис. 4. Диалог выбора динамических свойств объекта. |

Все динамические связи можно разделить на две группы: Touch Links (левая колонка) и Display Links (три колонки справа). С помощью свойств Touch Links выполняется какой - либо ввод в систему. Свойства Display Links осуществляют вывод информации на экран дисплея.   
  
Нажатие на любую клавишу диалога (рис. 4) вызывает появление нового диалога для определения соответствующего свойства объекта. Количество диалогов соответствует количеству динамических свойств (кнопок) диалога выбора. Все диалоги различны, но большинство из них имеет общие характеристики:

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

На рис.5 приведен диалог для определения свойств объекта (кнопки), управляющего значением дискретной переменной.

|  |
| --- |
| Рис.5. Диалог определения свойств кнопки. |

Завершение работы с диалогом производится нажатием кнопки Ok. Если переменная поля Tagname была ранее определена в словаре переменных данного приложения, пользователь возвращается в диалог выбора динамических свойств объекта (рис. 4). Можно либо продолжить определение других динамических свойств для данного объекта, либо, нажав Ok, вернуться на поле разработки окна приложения.   
Сложные объекты.

* Символ - это некоторая комбинация простых объектов, которые обрабатываются как один объект. Любое изменение статических или динамических свойств символа влияет на все составляющие символа. Например, если создать символ "насос" из двух кругов и двух прямоугольников и присвоить ему динамическое свойство Fill Color (цвет заполнения), то это свойство будет распространяться на все четыре простых объекта.   
  Различные объекты символа могут иметь разные значения одного и того же свойства, если они были присвоены этим объектам до объединения в символ. Bitmap - объекты, кнопки, компоненты не могут быть включены в состав символа.
* Компонент - это совокупность двух или более объектов, символов или других компонентов, образующих единый элемент. Они создаются путем выбора двух и более объектов, символов или компонентов и последующего запуска команды Arrange/Make Cell. Компоненты реализуют пространственную взаимосвязь между составляющими их графическими элементами. Каждая составляющая компонента может иметь свои собственные динамические свойства. Компоненты используются для таких виртуальных устройств, как панель управления контроллером, движковый регулятор и т. д.   
  Компонент не может менять свой размер, ему нельзя присваивать динамические свойства (внутри компонента есть объекты и символы со своими динамическими свойствами). Нельзя изменять и статические свойства (внешний вид). Для изменения статических и динамических свойств компонента его надо разобрать на составные части командой Arrange/Break Cell. Однако компоненты можно дублировать, копировать, вставлять, выравнивать, перемещать и т. д.

Мастер-объект - это предварительно созданный компонент с определенными статическими и динамическими свойствами, находящийся в библиотеке мастер-объектов (Wizards) и доступный для многократного применения. Но, в отличие от компонента, динамические свойства которого настраиваются для каждой составляющей отдельно до объединения в компонент, динамические свойства мастер-объекта быстро настраиваются с помощью специализированного диалога. Другими словами, фирма Wonderware провела большую работу и создала огромное количество мастер-объектов (несколько тысяч), определив для каждого из них механизм быстрой настройки статических и динамических свойств. Все эти мастер-объекты разделены на большое количество групп и размещены в соответствующей библиотеке. Доступ к ней осуществляется нажатием иконки Wizard в интерфейсе WindowMaker, что вызывает появление на экране диалога Wizard Selection (Выбор мастер-объекта. В левой части диалога - список групп мастер-объектов, включающий такие категории, как Buttons (кнопки), Sliders (ползунковые регуляторы), Switches (переключатели) и т. д.   
В правой части диалога приведены все мастер-объекты выбранной в данный момент группы. Двойной щелчок по требуемому мастер-объекту возвращает пользователя в окно разработки приложения. Курсор принимает форму уголка с символом . Наконец, щелчок мыши на свободном месте окна приводит к появлению мастер-объекта в окне приложения. Для его конфигурирования (определения динамических свойств) следует дважды щелкнуть на мастер-объекте.

|  |  |
| --- | --- |
|  | Например, двойной щелчок по кнопке Momentary Button (кнопка запуска), предварительно вставленной в окно приложения, выводит на экран диалог конфигурирования этой кнопки (рис.6). |

Достаточно ввести имя дискретной переменной, желаемый текст на кнопке, отметить несколько опций и нажать Ok. Инструмент Bitmap инструментальной панели рисования позволяет копировать и встраивать в приложение InTouch растровые объекты (совокупность точек). С помощью него создается "контейнер" для последующей вставки объекта из папки обмена Windows либо файлов с расширением .BMP, .JPG, .PCX, .TGA. Для WindowMaker растровое изображение является единым объектом. Невозможно ни анимировать его отдельные части, ни вставлять Bitmap - объекты в символы (можно вставлять в компоненты). Такой объект можно развернуть на рабочем поле на 90, 180, 270, 360 градусов, а также определить для него цвет "прозрачности", чтобы через него можно было видеть и другие объекты.   
· Тренды. InTouch предлагает пользователю два сложных объекта типа тренд: тренд реального времени и исторический (архивный) тренд. Эти объекты позволяют отображать в виде графиков значения данных реального времени (4 пера) и архивных данных (8 перьев). Оба типа трендов создаются при использовании специальных инструментов панели рисования окна WindowMaker с последующим конфигурированием. Подробная информация по созданию и конфигурированию трендов будет приведена в соответствующей главе.   
  
Подводя итог описанию графических средств пакета InTouch, следует отметить, что фирма Wonderware в этом плане предлагает потребителю хороший набор возможностей:

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

# Организация взаимодействия с контроллерами

Современные SCADA - системы не ограничивают выбора аппаратуры нижнего уровня (контроллеров), так как предоставляют большой набор драйверов или серверов ввода/вывода и имеют хорошо развитые средства создания собственных программных модулей или драйверов новых устройств нижнего уровня.   
  
Для подсоединения драйверов ввода/вывода к SCADA - системе в настоящее время используются следующие механизмы:

* ставший стандартом de facto динамический обмен данными (DDE);
* собственные протоколы фирм-производителей SCADA - систем, реально обеспечивающие самый скоростной обмен данными;
* новый OPC - протокол, который, с одной стороны, является стандартным и поддерживается большинством SCADA - систем, а с другой стороны, лишен недостатков протоколов DDE.

Изначально протокол DDE применялся в первых человеко - машинных интерфейсах в качестве механизма разделения данных между прикладными системами и устройствами типа ПЛК (программируемые логические контроллеры). Для преодоления недостатков DDE, прежде всего для повышения надежности и скорости обмена, разработчики предложили свои собственные решения (протоколы), такие как AdvancedDDE или FastDDE - протоколы, связанные с пакетированием информации при обмене с ПЛК и сетевыми контроллерами. Но такие частные решения приводят к ряду проблем:

* для каждой SCADA - системы пишется свой драйвер для поставляемого на рынок оборудования;
* в общем случае, два пакета не могут иметь доступ к одному драйверу в одно и то же время, поскольку каждый из них поддерживает обмен именно со своим драйвером.

Основная цель OPC стандарта (OLE for Process Control) заключается в определении механизма доступа к данным с любого устройства из приложений. OPC позволяет производителям оборудования поставлять программные компоненты, которые стандартным способом обеспечат клиентов данными с ПЛК. При широком распространении OPC - стандарта появятся следующие преимущества:

* OPC позволят определять на уровне объектов различные системы управления и контроля, работающие в распределенной гетерогенной среде;
* OPC - устранят необходимость использования различного нестандартного оборудования и соответствующих коммуникационных программных драйверов;
* у потребителя появится больший выбор при разработке приложений.

С OPC - решениями интеграция в гетерогенные (неоднородные) системы становится достаточно простой. Применительно к SCADA-системам OPC серверы, расположенные на всех компьютерах системы управления производственного предприятия, стандартным способом могут поставлять данные в программу визуализации, базы данных и т. п., уничтожая, в некотором смысле, само понятие неоднородной системы.

## Аппаратная реализация связи с устройствами ввода/вывода

Для организации взаимодействия с контроллерами могут быть использованы следующие аппаратные средства:

* COM - порты.   
  В этом случае контроллер или объединенные сетью контроллеры подключаются по протоколам RS-232, RS-422, RS-485.
* Сетевые платы.   
  Использование такой аппаратной поддержки возможно, если соответствующие контроллеры снабжены интерфейсным выходом на Ethernet.
* Вставные платы.   
  В этом случае протокол взаимодействия определяется платой и может быть уникальным. В настоящее время предлагаются реализации в стандартах ISA, PCI, CompactPCI.

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

## Серверы ввода/вывода в InTouch

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

## Поддерживаемые коммуникационные протоколы

DDE (Dynamic Data Exchange - динамический обмен данными) представляет собой коммуникационный протокол, разработанный компанией Microsoft для обмена данными между различными Windows - приложениями. Этот протокол реализует взаимосвязи типа клиент - сервер между двумя одновременно исполняющимися программами.   
В InTouch поддерживается также пакетированный DDE - обмен - FastDDE. Применение последнего заметно повышает эффективность и производительность обмена данными благодаря уменьшению общего количества DDE - пакетов, которыми клиент и сервер обмениваются между собой. Но принципиальные недостатки, связанные с надежностью и зависимостью от количества загруженных в текущий момент приложений Windows, остались. Необходимость в появлении более совершенного технологичного протокола созрела! Но следует отметить, что отказ от DDE-механизма происходит не мгновенно хотя бы потому, что в мире наработано большое количество DDE - серверов.   
С целью расширения возможностей стандартного протокола DDE на локальную сеть компания Wonderware предложила NetDDE. Он позволяет приложениям, запущенным на объединенных в локальную сеть компьютерах, вести DDE - обмен. Позднее NetDDE лицензируется компанией Microsoft и поставляется в дистрибутивном пакете Windows. Следует отметить и то, что NetDDE допускает обмен информацией между приложениями на IBM PC и приложениями на машинах другого типа с операционной системой VMS или UNIX. Компания Wonderware предлагает и инструментальные средства для разработки DDE-серверов, в том числе и для не-Windows-платформ.   
Протокол SuiteLink был специально разработан фирмой Wonderware для того, чтобы удовлетворить таким требованиям, как целостность данных, высокая производительность и простота диагностики. В основе протокола SuiteLink лежит протокол TCP/IP. SuiteLink не является заменой протоколам DDE, FastDDE и NetDDE. Новый протокол разработан для поддержания быстродействующих промышленных систем и обладает следующими характеристиками:

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

Для реализации функций OPC - клиента Wonderware предлагает OPCLink - сервер, преобразующий OPC в SuitLink - протокол.   
В материалах, предложенных компанией Wonderware, отмечается, что большинство реализованных OPC-серверов создают для каждого подключаемого к серверу клиента новый канал связи или нить. Для текущей обработки каждого клиента сервер должен переключаться между нитями. Каждая нить использует DCOM (Distributed Component Object Model) для организации обмена данными, и DCOM также управляет переключением нитей. В итоге возможна достаточно низкая производительность в сети.   
Тесты, проведенные фирмой Wonderware, показали, что при обслуживании OPC-сервером 7 клиентов (при передаче 4 целых чисел в режиме обновления) сервер на 95% занимал ресурсы CPU. Это означает, что ресурсы компьютера практически целиком были заняты переключением нитей и DCOM- процедурами.   
Поэтому на текущем этапе параметры производительности протокола SuiteLink превосходят параметры DCOM. Поставляемый в комплекте FactorySuite (Wonderware) OPCLink Server обеспечивает прием информации с OPC- сервера и передачу ее по протоколу SuiteLink в SCADA - систему InTouch и наоборот. Именно OPCLink Server рекомендуется устанавливать на одном узле с OPC- сервером, чтобы для сетевых передач использовался SuiteLink- протокол, а не DCOM (рис.7).

|  |
| --- |
| Рис. 7. Использование SuiteLink - протокола в SCADA - системах. |

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

## Особенности адресации в InTouch

В InTouch вышеуказанные механизмы положены в основу обмена данными между приложениями InTouch и DDE и SuiteLink - серверами, которые, в свою очередь, связаны коммуникационными каналами с устройствами нижнего уровня (контроллерами).   
  
Так как InTouch предназначен для разработки и поддержания интерфейса сбора данных и диспетчерского управления (рис.8), среда исполнения WindowViewer при взаимодействии с контроллерным уровнем выступает, как правило, в роли приложения - клиента (узел View), запрашивающего данные у приложения - сервера (I/O Server).

|  |
| --- |
| Рис.8. Обмен данными между InTouch - приложением и технологическим процессом. |

Через сервер ввода/вывода InTouch - приложение имеет возможность читать данные из контроллера или писать данные в него. Процесс обмена информацией InTouch - приложения с контроллером можно представить следующей схемой   
Здесь и встает один из главных вопросов организации обмена с серверами ввода/вывода: каким образом обеспечить клиенту доступ к запрашиваемой им информации?   
Для организации обмена с приложением определяются каналы обмена или каналы доступа, характеризующиеся следующими параметрами:

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

Имя приложения - это имя программы Windows, которая выполняет функции DDE, FastDDE, SuiteLink - серверов. Имя группы данных (топика) определяется при конфигурировании сервера на прием или передачу группы данных, которыми сервер будет обмениваться с контроллером или объединенными в сеть контроллерами. Определенные параметры группы (топика) зависят от конкретного сервера (поэтому рекомендуется изучать документацию и справочную систему выбранного сервера). Например, при использовании Modbus - сервера, позволяющего обеспечить взаимодействие с контроллером Modicon Micro 984 PLC, в качестве имени приложения (Application Name) должен быть Modbus, в качестве имени группы или топика (Topic Name) вводится любое имя (текстовая строка), но среди необходимых параметров группы из списка выбирается имя контроллера Modicon 984 PLC. А в качестве имени элемента (Item Name) следует выбирать название конкретного регистра контроллера (например, 40001 для контроллера Modicon Micro 984). Чтобы узнать правильный синтаксис имени элемента, необходимый для конкретных PLC, нужно обратиться к руководству по соответствующему серверу.   
  
Определены все компоненты коммуникационного канала. С учетом введенных понятий схема обмена информацией для рассмотренного выше примера будет выглядеть следующим образом (рис.9).

|  |
| --- |
| Рис. 9. Обмен информацией на примере Modbus - сервера. |

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

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

Схемы, приведенные на рис. 9, интерпретируют стандартный обмен информацией между узлом (приложением) View и контроллером (ПЛК) в режиме сбора данных и управления. В этом режиме, как уже было сказано выше, приложение View - клиент по определению.

## Обмен данными с другими приложениями

Но приложения InTouch могут взаимодействовать не только между собой, но и с другими Windows - приложениями. Одним из известных примеров такого приложения является Microsoft Excel. InTouch - приложение может считывать и записывать какие - либо значения в любую клетку открытой в Excel электронной таблицы. Аналогично и программа Excel может читать и записывать информацию в базу данных InTouch - приложения. Данный механизм обеспечивает одновременное обновление данных в одном приложении при изменении их значений в другом.   
  
Если клиентом (приложением, запрашивающим информацию) по - прежнему является узел View, то Excel - это приложение, поставляющее информацию (сервер). В качестве группы или топика (Topic) тогда будет выступать имя таблицы Excel, а элемент обмена информацией - ячейка в таблице Excel (табл.2.1, вариант 1).   
  
Когда клиентом является приложение Excel, а сервером - приложение View, группой в этом случае всегда является словарь переменных InTouch (база данных) с именем Tagname. Элементом обмена будет элемент базы данных - имя переменной (табл.2.1, вариант 2).

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Таблица 2.1.   |  |  |  |  | | --- | --- | --- | --- | | Приложение-клиент | Приложение-сервер | Группа | Элемент | | View | Excel | Sheet1.XLS | R1C1 | | Excel | View | Tagname | R\_Level | |

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

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

## Определение имени доступа в словаре переменных InTouch

В InTouch - приложениях вся информация о переменных приложения хранится в Tagname Dictionary (Словарь переменных). Это не что иное, как база данных реального времени - один из центральных компонентов InTouch.   
При определении переменной в базе данных InTouch запрашивает определенную информацию о каждой переменной, например, имя переменной, ее тип, имя доступа и т. д.   
В пакете InTouch используется два базовых типа переменных - Memory (внутренние) и I/O (переменные ввода/вывода).   
Переменные типа Memory могут быть использованы для создания различных системных констант, моделирования элементов системы управления и в вычисляемых переменных, доступных другим Windows - программам.   
Все переменные, которые получают или передают свое значение другой Windows - программе, должны иметь тип ввода/вывода (I/O). В эту категорию попадают переменные, которые посредством канала доступа (Access Name) принимают или отправляют данные из/в серверов ввода/вывода, других приложений InTouch, других программ Windows.   
Определение новой переменной в базе данных InTouch, как и просмотр, и модификация атрибутов уже существующих переменных, производится в диалоге Tagname Dictionary (рис.10). Доступ к этому диалогу осуществляется командой Speсial/Tagname Dictionary в окне среды разработки WindowMaker или двойным щелчком по иконке Tagname Dictionary в окне Application Explorer.

|  |
| --- |
| Рис. 10. Диалог Tagname Dictionary (Словарь переменных). |

Поля Tagname и Comment предназначены для ввода имени переменной и соответствующего комментария. По умолчанию включена опция Read/Write (чтение/запись). Можно отметить и опцию Read Only, если в процессе исполнения WindowViewer должен только читать значение переменной.

В любое время в режиме проектирования можно открыть список переменных приложения щелчком по кнопке Select для выбора соответствующей переменной, просмотра списка или модификации атрибутов. Диалог Select Tag (выбор переменной) представлен на рис.11.

|  |
| --- |
| Рис. 11. Диалог Select Tag (выбор переменной). |

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

Группа алармов (Alarm group, рис.11) для переменной определяется в диалоге, вызываемом нажатием кнопки Group диалога Tagname Dictionary. Все, что касается алармов, рассматривается в соответствующем разделе ниже.

Выбор типа переменной осуществляется в диалоге Tag Types (тип переменной, рис. 12), вызываемом на экран нажатием кнопки Туре диалога Tagname Dictionary.

|  |
| --- |
| Рис. 12. Диалог Tag Types (тип переменной). |

В этом диалоге представлен полный список основных типов переменных InTouch. Выбор завершается отметкой соответствующей опции и щелчком по Ok.   
После выбора типа переменной программа возвращает пользователя в диалог Tagname Dictionary (Словарь переменных). При этом будет открыт и дополнительный диалог подробного описания переменной, содержание которого зависит от выбранного типа. Кнопка Access Name (имя доступа) используется для определения канала обмена (канала доступа) с сервером, с которым будет связана описываемая переменная. Имя доступа Access Name определяется именем узла, именем приложения и именем группы или топика. Имя топика должно совпадать с соответствующим именем, заданным при конфигурировании DDE, SuiteLink-сервера. Имя элемента, как компонента многоуровневого адреса, определяется в поле Item (рис.13).   
В распределенных системах InTouch имя доступа может быть определено либо как локальный адрес, либо как глобальный.   
Локальные адреса используются в том случае, когда View - узлы имеют свои серверы ввода/вывода. На рис. 13 узлы исполнения (View - узлы), каждый со своей копией одного и того же приложения, ссылаются на свои собственные источники данных ввода/вывода (серверы ввода/вывода).

|  |
| --- |
| Рис. 13. Сеть View – узлов с собственными серверами ввода/вывода. |

Поэтому при определении канала доступа к информации ввода/вывода достаточно трехуровневого адреса (Application - приложение, Topic - объект, Item - элемент). Имя узла (Node) в этом случае опускается. Щелчок по кнопке Access Name (рис.2.3.8) вызывает на экран одноименный диалог. Этот диалог предназначен для определения нового канала доступа (кнопка Add), модификации существующего (Modify) или удаления (Delete). Щелчок по кнопке Add вызывает диалог определения нового канала доступа. В качестве имени (канала) доступа (Access Names) рекомендуется выбирать имя группы или топика (Topic Name). Следует подчеркнуть, что поле Node Name (имя узла) оставлено пустым. Щелчок по кнопке Ok возвращает пользователя в диалог Access Names (имена доступа) с определенным именем доступа.  
Глобальные адреса источников данных ввода/вывода позволяют нескольким View - узлам обращаться к одному и тому же серверу ввода/вывода. Такой подход предоставляет возможность отказаться от нескольких серверов ввода/вывода, однако менее защищен от отказов (рис.14).

|  |
| --- |
| Рис.14. Архитектура с двумя View - узлами и сервером ввода/вывода. |

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

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

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

# Тренды в SCADA - системах

Графическое представление значений технологических параметров во времени способствует лучшему пониманию динамики технологического процесса предприятия. Поэтому подсистема создания трендов и хранения информации о параметрах с целью ее дальнейшего анализа и использования для управления является неотъемлемой частью любой SCADA - системы.

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

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

## Тренды в InTouch

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

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

## Архивирование (регистрация) значений переменной

При работе системы в режиме WindowViewer (среда исполнения) InTouch может производить запись значений переменных в регистрационный файл. Для того, чтобы архивирование переменной выполнялось, необходимо включить опцию Log Data (регистрация данных) при определении переменной в диалоге Tagname Dictionary.   
Запись в регистрационный файл производится всякий раз при изменении переменной на величину, превышающую порог для архивирования (Log Deadband), и по умолчанию один раз в час, если значение переменной за это время не изменилось. Поле Log Deadband находится в диалоге детального описания целой или вещественной переменной.  
Чтобы значения переменных, для которых опция Log Data разрешена, записывались в регистрационные файлы, необходимо общее разрешение глобальной функции регистрации. Его задают в диалоге Historical Logging Properties (параметры архивирования, рис. 15), который вызывается на экран командой Special/Configure/Historical Logging. В этот диалог можно также войти из окна Application Explorer.

|  |
| --- |
| Рис.15. Диалог Historical Logging Properties. |

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

## Отображение трендов

Тренды реального времени являются динамическими объектами. Они позволяют выводить изменения значений переменных, как только они происходят для любой конкретной переменной или для выражения, которое содержит одну или несколько переменных. Данные будут появляться в окне тренда и двигаться справа налево.   
Чтобы создать тренд реального времени, необходимо:

* выбрать инструмент тренд реального времени в панели инструментов WindowMaker;
* щелкнуть в окне, затем переместить мышь по диагонали и сформировать прямоугольник необходимого размера;
* отпустить кнопку мыши, что вызовет появление тренда реального времени в окне (рис.16).

|  |
| --- |
| Рис.16. Объект "тренд реального времени". |

При создании тренда реального времени настройки его конфигурации устанавливаются по умолчанию (настройки предыдущего тренда).

Для конфигурирования тренда реального времени следует либо дважды щелкнуть на созданном объекте, либо, предварительно выбрав объект, запустить команду Special/Animation Links. На экране появится диалог Real Time Trend Configuration (конфигурирование тренда реального времени). Среди настроек этого диалога можно отметить диапазон времени, охватываемый трендом (Time Span), частоту вывода значение переменной (Interval), разрешение сетки по большим и малым делениям горизонтальной и вертикальной осей (Time Division, Value Division), цвета фона и рамки графика (Color). Конфигурирование перьев тренда включает выбор имени переменной или выражения, цвета и толщины линии для каждого пера (поле Expression). Для повышения производительности системы следует отметить опцию Only update when in memory (обновлять, когда в памяти). В этом случае обновление данных тренда будет производиться только в моменты, когда окно с трендом отображается на дисплее (находится в RAM). Есть и другие способы повышения производительности при работе с трендами реального времени (уменьшение толщины линии графика, уменьшение частоты выводы значений переменной). Например, если установлен диапазон времени (Time Span) в 30 минут, а частота вывода - 2 секунды, то число измерений, которые нужно провести за каждые 30 минут, будет равно 900 (30 \* 60/2 = 900). При частоте выводе в 5 секунд число измерений существенно уменьшается: 30 \* 60/5 = 360. Исторические (архивные) тренды не являются динамическими. Они обеспечивают "снимок" состояния данных за прошедшее время, то есть по архивным данным. В отличие от трендов реального времени исторические тренды обновляются только по команде - при запуске скрипта, изменении значения выражения или нажатии оператором соответствующей кнопки. При конфигурировании архивного тренда можно создать "визиры" (ползунки, бегунки), с помощью которых удобно получить значения всех отображаемых переменных на один и тот же момент времени. Бегунки архивного тренда представляют собой позиционные индикаторы на временной оси, положение которых определяет объем извлекаемых данных. Связав объект "движковый регулятор" с полем бегунка, можно осуществлять перемещение вдоль архивного тренда. Кроме того, имеются функции вычисления среднего, минимального и максимального значений в определенном бегунком положении. Можно создать правый и левый бегунки и производить обработку данных кривой, расположенной между бегунками. Вычисляются следующие величины: среднее, минимальное, максимальное, отношение мин/макс и стандартное отклонение. В зависимости от положения бегунков на оси можно реализовать и другие функции (увеличение и уменьшение заключенной между бегунками области графика).

Благодаря системе распределенных архивов на один и тот же график можно выводить информацию из нескольких баз данных.

Все сказанное выше о механизме создания тренда реального времени инструментом Real Time Trend в среде разработки WindowMaker и о его последующем конфигурировании можно отнести и к архивному тренду, создаваемому инструментом Historical Trend среды разработки.

Предлагаемый ниже способ создания и конфигурирования архивного тренда предполагает использование мастер-средств библиотеки Wizard.

Нажатие кнопки выбора мастер-средств в панели инструментов вызывает появление на экране диалога Wizard Selection (выбор мастер-средств).   
После выбора из предложенного набора мастер-средств Hist Trend with Scooters (архивный тренд с бегунками) и щелчка по Ok программа возвращает пользователя в среду разработки. Курсор мыши при этом примет форму вставки . Последующий щелчок мыши на предполагаемом месте нахождения создаваемого объекта выводит на экран архивный тренд (рис.17). Объекты этого типа ведут себя аналогично любым другим объектам, то есть их можно перемещать, масштабировать и т. д.

|  |
| --- |
| Рис.17. Объект "архивный тренд". |

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

|  |
| --- |
| Рис.18. Диалог конфигурирования архивного тренда. |

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

Чтобы добавить в тренд функции масштабирования и перемещения или элементы управления перьями, следует использовать панели Zoom/Pan и Trend Pen Legend (рис.16), соответственно. Для того, чтобы эти компоненты работали совместно, они должны иметь одинаковые имена (Hist Trend).

## Изменение параметров архивных трендов в режиме исполнения

При управлении в режиме реального времени оператор анализирует архивную информацию. Объем информации, ее временные диапазоны, объем статистических данных, необходимые для принятия решения по управлению технологическим процессом, заранее не известны. Поэтому оператор должен иметь возможность менять настройки архивных трендов, не выходя из режима Runtime. В InTouch такая возможность существует.   
Для этого следует включить опцию Allow runtime changes (разрешить изменения во время исполнения) в диалоге конфигурирования архивного тренда (в книге не показан).   
Теперь в режиме WindowViewer щелчок на архивном тренде будет вызывать на экран диалог изменения параметров архивного тренда (Historical Trend Setup). В этом диалоге можно определить дату и время начала архивного тренда (поле Chart Start), его временной диапазон (Chart Length), присвоить перьям цвет и имена переменных, выбирая их из словаря.   
Архивный тренд может выводиться в одном из трех возможных режимах:

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

Выбор режима производится в поле Display Mode.

## Система распределенных архивов

В InTouch имеется система распределенных архивов, обеспечивающая поиск архивных данных в любом InTouch - приложении. Данная система расширяет возможности стандартных архивов InTouch, позволяя одновременно получать информацию из нескольких удаленных баз данных, которые в этом случае называются провайдерами архивов.   
Одновременно можно обращаться к восьми провайдерам (по одному на каждое перо). Каждый узел, выполняющий функцию регистрации, может писать только в один архив.   
Система, приведенная на рис.19, имеет два провайдера архивов. Левый провайдер регистрирует информацию только из узла, расположенного слева внизу. Правый провайдер регистрирует информацию из узла, расположенного справа вверху. Остальные три узла (вверху слева) лишь используют архивные данные. Читать информацию из архивных файлов может каждый из узлов системы.

Создание такой системы предполагает следующие действия:

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

|  |
| --- |
| Рис. 19. Распределенная система архивов. |

# Встроенные языки программирования

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

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

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

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

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

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

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

Каждая из функций во встроенном языке выполняется в синхронном или асинхронном режиме. В синхронном режиме выполнение следующей функции не начинается до тех пор, пока не завершилось исполнение предыдущей. При запуске асинхронной функции управление переходит следующей, не дожидаясь завершения исполнения предыдущей функции.   
В связи с этим возникает несколько вопросов. С каким приоритетом исполняется каждый из фрагментов, допускается ли рекурсия при обработке событий и если да, то каков уровень вложенности? В SCADA - системах уровень вложенности пока не стандартизован, но оговаривается особо в рамках каждой из них.

## Скрипты в InTouch

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

## Типы скриптов

В InTouch различают несколько типов скриптов:

* Application Scripts (скрипты уровня приложения) относятся ко всему приложению и используются для запуска других приложений, имитации технологических процессов, вычисления значений переменных и т.д.
* Window Scripts (скрипты уровня окна) связываются с конкретным окном.
* Key Scripts (клавишные скрипты) привязываются к какой-либо клавише или комбинации клавиш клавиатуры. Это может быть полезным при создании каких-либо глобальных для всего приложения функций (возврат в главное окно, окончание сеанса работы с приложением и т. д.).
* Touch Pushbutton Action Scripts (скрипты, запускаемые кнопками) очень похожи на клавишные скрипты и связываются с объектами, которые будут использоваться в качестве исполнительных кнопок. Эти скрипты запускаются при каждом нажатии на объект-кнопку.
* Condition Scripts (скрипты по изменению логического выражения) связываются с логической переменной или выражением, которое будет принимать значения либо "истина", либо "ложь". Логические скрипты могут содержать в себе и аналоговые переменные.
* Data Change Scripts (скрипты по изменению данных) связываются либо с переменной, либо с полем переменной. Эти скрипты исполняются только один раз, когда значение переменной либо поля меняется на величину, превышающую значение допуска, заданного в словаре переменных.
* ActiveX Event (скрипты событий ActiveX) предназначены для поддержки механизма реакции на события в ActiveX - объектах. С каждым событием может быть связан один скрипт типа ActiveX Event, запускающийся в WindowViewer во время исполнения приложения.
* Quick Function - скрипты, которые могут вызываться из других скриптов и использоваться в выражениях при определении динамических свойств объектов.

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

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

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

## Встроенные функции

В пакете InTouch имеется набор встроенных функций, которые могут быть связаны с командами или использованы в скриптах для выполнения самых различных задач.   
  
Все встроенные функции разбиты на четыре группы:   
- String... - для обработки различных символьных строк и переменных;   
- Math... - математические функции;   
- System... - системные функции;   
- Misc... - функции для работы с алармами распределенных систем, трендами, печатью и др.   
· Вызов списка функций группы осуществляется нажатием соответствующей клавиши. Например, щелчок по клавише String... редактора скриптов вызывает появление диалога Choose function (выбор функции) со списком строковых функций.

Описание некоторых функций этого списка приведено в табл. 5.1.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Функция | Описание | | StringFromIntg() | Возвращает символьное представление целого аргумента в указанной системе счисления | | StringFromReal() | Возвращает символьное представление вещественной величины либо в формате с плавающей запятой, либо в экспоненциальном формате | | StringLen() | Возвращает длину указанной строки | | StringToIntg() | Преобразует символьное представление целого числа во внутренний формат | | StringUpper() | Преобразует все символы исходной строки в нижнем регистре в верхний регистр | | Text() | Осуществляет форматированный вывод указанной целой или вещественной переменной в соответствии со строкой форматирования |   Таблица 5.1. |

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

StringFromReal(Number,Precision,Type);

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

Например,

функция StringFromReal(263.365, 2, "f") возвращает "263.36";   
функция StringFromReal(263.365, 2, "e") возвращает "2.63e2";   
функция StringFromReal(263.55, 3, "E") возвращает "2.636E2".   
Функция Text имеет два аргумента: Text(Analog\_Tag, "Format\_Text");

- Analog\_Tag - вещественное или целое число;   
- Format\_Text - формат преобразования.

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

- при Analog\_Tag = 66 функция возвращает 66.00;   
- при Analog\_Tag =22.269 функция возвращает 22.27;   
- при Analog\_Tag =9.999 функция возвращает 10.00.

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

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

* Системные функции делятся на две категории: файловые (File) и для работы с Windows - приложениями (Info).

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

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Функция | Описание | | FileCopy() | Копирует исходный файл в файл-приемник | | FileReadFields() | Возвращает очередную запись данных из CSV - файла | | FileReadMessage() | Возвращает указанное количество байтов (или всю строку) из указанного файла | | FileWriteFields() | Сохраняет в CSV - файле запись данных, состоящую из разделенных запятыми величин | | InfoDisk() | Возвращает информацию об указанном локальном или сетевом диске | | InfoFile() | Возвращает информацию об указанном файле или подкаталоге компьютера или сетевого устройства | | InfoTouchAppDir() | Возвращает имя текущего каталога InTouch - приложения |   Таблица 5.1. |

Остальные аргументы файловых функций не поддаются типизации и различны для каждой функции.   
  
Например, функция FileReadFields имеет четыре аргумента и следующий синтаксис:

FileReadFields(Filename,FileOffset,StartTag,NumberOfFields);

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

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

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

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Функция | Описание | | almAckDisplay() | Подтверждает только те алармы, которые в текущий момент видны в окне отображения алармов | | almAckSelect() | Подтверждает алармы, отмеченные оператором в окне отображения алармов | | almShowStats() | Выводит панель статистики объекта отображения алармов |   Таблица 5.3.1. |

Первым аргументом всех встроенных функций алармов является ObjectName (имя объекта алармов). Часто в роли одного из аргументов выступает Comment (комментарий). Например, функция almAckSelect имеет следующий синтаксис: almAckDisplay(ObjectName,Comment); .   
  
Функции, название которых начинается с HT, используются только с архивными трендами. Примеры таких встроенных функций - в табл.5.3.2.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Функция | Описание | | HTGetPenName() | Возвращает имя переменной, связанной в текущий момент с указанным пером указанного тренда | | HTGetValue() | Возвращает значение указанного типа, вычисляемого для указанного пера в пределах всего тренда | | HTScrollLeft() | Устанавливает в качестве начала графика более раннее время. Визуально происходит прокрутка тренда влево | | HTSetPenName() | Связывает перо тренда с указанной переменной | | HTZoomIn() | Масштабирует существующий тренд путем задания новых времени начала и охватываемого интервала времени |   Таблица 5.3.2. |

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

- HTGetPenName(Hist\_Tag, UpdateCount, PenNum);   
- HTGetValue(Hist\_Tag,UpdateCount,PenNum,ValType\_Text);   
- HTScrollLeft(Hist\_Tag,Percent);   
- HTSetPenName(Hist\_Tag,PenNum,Tagname);   
- HTZoomIn (Hist\_Tag,LockString).

Первый аргумент всех встроенных функций для работы с трендами - Hist\_Tag (имя тренда). Из других аргументов следует отметить PenNum (номер пера тренда), ValType\_Text (строка, указывающая тип возвращаемого значения), Tagname (новое имя пера).   
  
Функции, название которых начинается с wc (табл.5.3.3), используются с управляющими объектами окна (простые списки, текстовые окна, ниспадающие списки и т. д.)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Функция | Описание | | wcDeleteItem() | Уничтожает элемент с заданным порядковым номером как в простом, так и в ниспадающем списке | | wcInsertItem() | Вставляет указанное сообщение в список | | wcLoadText() | Заменяет содержимое текстового окна на новую информацию |   Таблица 5.3.3. |

Функции этой подгруппы также могут иметь до четырех аргументов:

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

Первый аргумент всех встроенных функций этой подгруппы - ControlName (имя управляемого окна). Часто в качестве аргумента используются ItemIndex (номер, соответствующий позиции элемента), MessageTag (строковое сообщение), Filrename (имя файла в формате ASCII).   
В рассматриваемой группе функций Miscellaneous следует отметить функцию PrintWindow, i?aaiacia?aiioю для печати окна. Ее синтаксис выглядит следующим образом:

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

где:

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

Вставка встроенных функций в скрипт производится щелчком по выбранной функции в списке функций. Она вместе со своими аргументами будет автоматически вставлена в текст скрипта в точку, указанную курсором. После этого можно отредактировать список аргументов.   
По окончании редактирования скрипта следует нажать кнопку Ok. При обнаружении в скрипте каких-либо ошибок на экран будет выведено соответствующее сообщение. В большинстве случаев курсор установится в ту позицию, которая привела к появлению ошибки. Прежде чем скрипт будет сохранен, все ошибки должны быть исправлены.

## Функции Quick Functions

Quick Functions - это скрипты, которые могут вызываться из других скриптов и использоваться в выражениях при определении динамических свойств объектов. Скрипты Quick Functions хранятся внутри того приложения, в котором они были созданы, и могут многократно использоваться в других скриптах InTouch.   
Наиболее часто эти функции используют в выражениях при определении динамических свойств объектов. Чем это вызвано? Дело в том, что длина выражения в поле Expression диалогов определения динамических свойств объектов должна быть не более 256 символов. Это относится к таким динамическим свойствам, как цвет линии, цвет заполнения, изменение высоты и ширины, вертикальное и горизонтальное перемещение, вертикальное и горизонтальное заполнение, видимость, мерцание, ориентация, блокировка.   
 Для ввода более длинных выражений можно воспользоваться функциями Quick Functions. При этом выражение в поле Expression должно содержать операторы CALL вызова функций Quick Functions, каждая из которых, в свою очередь, должна иметь в качестве последнего оператора RETURN для возврата результата в вызывающее выражение. Организованное таким образом выражение может содержать многие тысячи символов и быть сколь угодно сложным.   
Сохраненная функция Quick Functions может быть использована в любом другом скрипте или выражении.   
Quick Functions могут быть синхронными и асинхронными скриптами. Синхронные скрипты выполняются последовательно, в то время, как после запуска одного асинхронного скрипта может быть запущен другой (синхронный или асинхронный) скрипт. Это позволяет отделять исполняющиеся довольно долго операции (типа обращений к базам данных) от основной программы. Асинхронные скрипты не могут возвращать результаты. Поэтому в качестве скриптов Quick Functions, используемых в выражениях (Expression) для определения динамических свойств объектов, следует применять только синхронные скрипты.   
Создание скриптов Quick Functions осуществляется в диалоговом окне редактора Quick Functions. Вызов этого диалога на экран в окне WindowMaker производится в командой Special/Scripts с последующим нажатием на строке Quick Functions.

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

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

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

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

# Разработка графопостроителя в системе InTouch

Данный раздел посвящен разработке четырехканального графопостроителя визуализирующего данные, поступающие по DDE каналу с DDE сервера. В программе предусмотрена возможность масштабирования по каждому из каналов.

## Разработка DDE-сервера

Приложение, получающее данные из другого приложения по DDE и/или управляющее другим приложением с помощью команд через DDE является DDE-клиентом. В этом случае второе приложение является DDE-сервером. Рассмотрим проект DDE-сервера, выполненного на языке программирования Borland Delphi 6.

На рис.20 представлено окно DDE-сервера во время дизайна в среде Delphi



Рис. 20. Окно DDE-сервера на стадии проектирования в Delphi

Для построении DDE-сервера в Delphi имеются два объекта, расположенные на странице System Палитры Компонент - TDdeServerConv и TDdeServerItem. Обычно в проекте используется один объект TDdeServerConv и один или более TDdeServerItem. Для получения доступа к сервису DDE-сервера, клиенту потребуется знать несколько параметров : имя сервиса (Service Name) - это имя приложения (обычно - имя выполняемого файла без расширения EXE, возможно с полным путем); Topic Name - в Delphi это имя компоненты TDdeServerConv; Item Name - в Delphi это имя нужной компоненты TDdeServerItem. Назначение объекта TDdeServerConv - общее управление DDE и обработка запросов от клиентов на выполнение макроса.

Объект TDdeServerItem связывается с TDdeServerConv и определяет, что, собственно, будет пересылаться по DDE. Для этого у него есть свойства Text и Lines. (Text имеет то же значение, что и Lines[0].) При изменении значения этих свойств автоматически происходит пересылка обновленных данных во все приложения-клиенты, установившие связь с сервером.

При запуске приложения происходит выполнение процедуры TDDEServe.FormActivate:

procedure TDDEServe.FormActivate(Sender: TObject);

var nidata : TNotifyIconData;

begin

Application.ShowMainForm := False;

ShowWindow(Application.Handle, SW\_HIDE);

ShowWindow(Application.MainForm.Handle, SW\_HIDE);

with nidata do

begin

cbSize := SizeOf(TNotifyIconData);

Wnd := Self.Handle;

uID := 1;

uFlags := NIF\_ICON or NIF\_MESSAGE or NIF\_TIP;

uCallBackMessage := WM\_MYICONNOTIFY;

hIcon := Application.Icon.Handle;

StrPCopy(szTip,Application.Title);

end;

Shell\_NotifyIcon(NIM\_ADD, @nidata);

ru:=10;

end;

В этой процедуре приложение сворачивается в системный Tray, а форма становится невидимой. Окончание работы DDE-сервера вызывается путём нажатия левой или правой кнопкой мыши на иконке приложения в области системного Tray. Обработка этого события выполняется в процедуре TDDEServe.WMICON:

procedure TDDEServe.WMICON(var msg: TMessage);

begin

case msg.LParam of

WM\_RBUTTONDOWN,WM\_LBUTTONDOWN: close;

end;

end;

При этом, при закрытии окна приложения вызывается процедура TDDEServe.FormDestroy, в которой происходит удаление иконки из системного Tray:

procedure TDDEServe.FormDestroy(Sender: TObject);

var nidata : TNotifyIconData;

begin

with nidata do

begin

cbSize := SizeOf(TNotifyIconData);

Wnd := Self.Handle;

uID := 1;

end;

Shell\_NotifyIcon(NIM\_DELETE, @nidata);

end;

Работа приложения в целом строится посредством вызова процедуры TDDEServe.Timer1Timer по прерыванию таймера.

implementation

{$R \*.DFM}

uses ComObj, activex, ShellApi, shlobj, registry;

var

xsin: integer;

ru:real;

boolka:boolean;

procedure TDDEServe.Timer1Timer(Sender: TObject);

var LPTbyte: byte;

begin

xsin:=xsin+1;

if xsin>1000 then xsin:=xsin-1000;

DDEItem100.Text:=inttostr(5\*(xsin-20\*trunc(xsin/20))); //пилообразный сигнал

asm

mov dx,379h

in al,dx

and al,80h

mov LPTbyte,al

end;

DDEItem200.Text:=inttostr(LPTbyte\*100); //состояние линии LPT-порта

DDEItem300.Text:=inttostr(round(50+50\*sin(xsin/20)));

if (xsin/5)=trunc(xsin/5) then

if (ru<round(50+50\*sin(xsin/20))) then

begin

boolka:=true;

ru:=ru+20

end else

begin

boolka:=false;

ru:=ru-20

end;

if boolka then DDEItem400.Text:='100' else DDEItem400.Text:='0';

end;

Для обеспечения DDE-обмена задекларированы следующие константы:

‘DDEServer’ – имя сервиса (Service Name);

‘DDETopic’ – Topic Name;

‘DDEItem100’ – переменная обмена;

‘DDEItem200’ – переменная обмена;

‘DDEItem300’ – переменная обмена;

‘DDEItem400’ – переменная обмена.

## Разработка DDE - клиента

Основа человеко-машинного интерфейса в рамках InTouch - это иерархически взаимосвязанные анимированные сенсорные окна. Для создания нового окна выполним команду File/New Window... (Файл/Новое Окно). На экране появится диалоговое окно Window Properties (Свойства Окна), которое необходимо заполнить.

Здесь следует ввести только имя окна (поле Name) Scope. Ос­тальные поля и опции оставлены без изменений. Окно с указан­ными атрибутами появится на экране. Там же будет отображена и Панель Инструмен­тов InTouch - Tools, с которой предстоит интенсивно работать далее.

График представляет собой прямоугольную область с нанесенными координат­ными прямыми, на которой графически представляется изменение значения одной или нескольких переменных в течение времени. В пакете InTouch имеются объекты для ди­намического отображения значения переменной в реальном времени – графики реаль­ного времени (инструмент (Real-time Trend)), и, так называе­мые, аналитические кривые, которые строятся на основании архивных данных (инструмент (Historical Trend)). Для того, чтобы он появился в созданном нами окне, необходимо на Панели Инструмен­тов InTouch – Tools выбрать пункт Real – Time Trend и затем в окне приложения, удерживая нажатой левую кнопку мышки, придать графику необходимые размеры.

Панель настройки графиков вызывается двойным щелчком левой кнопки мышки по окну графика и состоит из четырех текстовых строк соответствующих графикам (Graph 1, Graph 2, Graph 3, Graph 4). Каждый график имеет независимые настройки масштаба и величины сдвига по вертикали, отображаемые на экране. Для этого используется компонент Analog Tagname Display (Wizard Selection → Value Displays → Analog Tagname Display). Ввод данных осуществляется не непосредственно в WindowViewer, а посредством компоненты Incr/Decr Buttons Up/Down (Wizard Selection → Buttons → Incr/Decr Buttons Up/Down). Нажатие на верхнюю или нижнюю стрелку приводит соответственно к увеличению или уменьшению значения переменной. Ограничение максимального и минимального значения указываются при декларации. Каждый компонент связан со своей переменной zoom1 – zoom4 (изменение масштаба графиков 1 – 4 соответственно) и sh1 – sh4 (изменение смещения графиков 1 – 4). Все переменные имеют тип Memory Integer. Для того, чтобы ввести новую переменную, необходимо описать ее в разделе Special/Tagname Dictionary/New. При этом необходимо указать ее имя в поле «Tagname:» и тип – в поле «Type:».

Для организации обмена данными через DDE интерфейс необходимо определить четыре (по числу каналов) переменные типа DDE Integer (Item1, Item2, Item3, Item4). Для этого сначала в разделе Special/DDE Access Names… необходимо нажать кнопку Add и в появившемся диалоговом окне указать имя приложения (DDE Application/Server Name), от которого будет производиться запрос данных, и имя группы/объекта (DDE Topic Name), содер­жащего требуемую информацию. В нашем случае качестве имени приложения используется имя DDEServer, имя объекта – DDETopic. Далее в разделе Special/Tagname Dictionary/New вводятся поочередно переменные типа DDE Integer. Название элемента (Item) для каждой переменной имеет различные имена: DDEItem100 – для Item1, DDEItem200 – для Item2, DDEItem300 – для Item3 и DDEItem400 – для Item4. Данная информация используется для определения DDE-переменной в Словаре Переменных InTouch.

Для того, чтобы запустить программу графопостроителя и начать DDE – обмен, необходимо включить DDE сервер (т. е. запустить файл Ddeserver.exe) и переключиться в окно InTouch - WindowViewer (нажатием кнопки Runtime! в правом верхнем углу окна InTouch - WindowMaker). В процессе работы InTouch WindowViewer автоматически выполнит все требуемые действия по установлению канала об­мена данными и обработке значений элемента.

Ниже представлен внешний вид программы графопостроителя в окне InTouch - WindowViewer отображающей в виде четырех графиков данные, полученные от программы DDE сервера и соответствующие им масштабирующие коэффициенты.

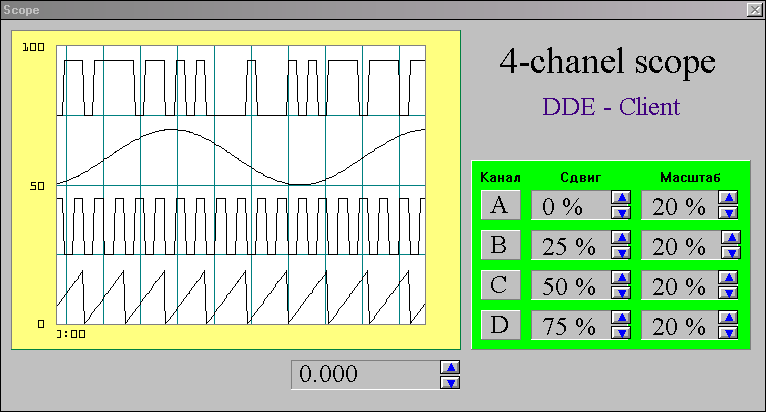


Рис. 21. Окно программы графопостроителя.

# Список литературы

1. Scada . ru - Публикации - SCADA - системы: взгляд изнутри

// URL: http://www.scada.ru/publication/book/preface.html

2. Кабаев С.В. Пакет программного обеспечения Intouch - система мониторинга и управления в объектах промышленной автоматизации

// URL: http://www.mka.ru/go/?id=40463&url=www.rtsoft.ru

**3. ТРЕЙС МОУД -** интегрированная SCADA- и softlogic-система для разработки АСУТП // URL: http://adastra.ru/ru/tm/tm5/

4. КузнецовА. Genesis for Windows – графическая scada-система для разработки АСУ ТП. // Современные технологии автоматизации.- 1997.- №3.