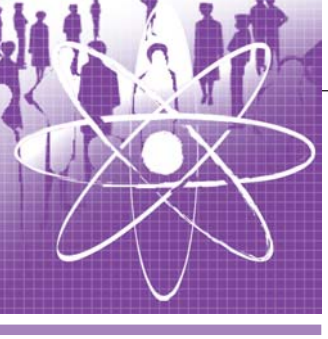

Где зарыт барсук?

Николай Горбунов

Рассматриваются средства программирования и способы построения распределённых АСУ ТП с использованием операционной системы QNX.

Китайская пословица гласит: «Лучше один плохой генерал, чем десять хороших». Древняя мудрость актуальна и по сей день — отсутствие централизованного управления и единой концепции порождает хаос и неразбериху, часто сводя на нет какую бы то ни было эффективность работы отдельных подсистем, и к системам автоматизации это относится в первую очередь. Когда вопрос построения единого информационного пространства предприятия перешел из области мифов в разряд насущных проблем, оказалось, что проблема эта лишь немногим проще той, с которой столкнулись строители Вавилонской башни, когда вроде и задача ясна, и решение существует, и строители есть, и материалы в наличии, а совместимость предусмотреть почему-то забыли. Казалось бы, достаточно квалифицированного системного интегратора — и вопрос решен?..

Однако системных интеграторов такого масштаба, специализирующихся на автоматизации всех уровней предприятия и способных предложить единое готовое решение для любого спектра задач, просто не существует в природе — слишком уж разнообразны потребности. Поэтому всё более актуальным становится вопрос создания открытых систем управления, позволяющих интегрировать между собой решения различных подзадач от различных производителей. Очевидно, такой подход дает как раз искомую «золотую середину», поскольку найти готовое комплексное решение гораздо сложнее (а зачастую его просто нет), чем собрать систему на месте из подходящих «кубиков», что, кстати, можно сделать как прибегнув к помощи системного интегратора, так и силами локального отдела АСУ.

Название статьи — не случайность. Слово «барсук» по-немецки пишется как Dachs. Цель данного материала — рассказать о линейке продуктов DACHS® (Distributed Automation, Control and Host System — «Распределенная вычислительная система автоматизации и управления»), разработанной и поставляемой немецкой компанией Steinhoff Automation & Fieldbus-Systems. Эта линейка содержит программные и аппаратные средства, позволяющие строить открытые распределенные системы управления, пригодные как для АСУ ТП, так и для других применений, где требуются быстрый цикл опроса, надежность передачи данных и богатые возможности интеграции с верхними уровнями, в том числе в гетерогенных вычислительных средах.

Концепция DACHS

Если говорить в двух словах, концепция DACHS строится на понятии расширенного виртуального ПЛК (soft-PLC). Целью введения такого понятия было сохранить удобство программирования в стандарте МЭК 61131-3, но обойти свойственные «реальным» ПЛК функциональные ограничения (о том, как это реализуется, пойдёт речь далее).

Под «расширенным» виртуальным ПЛК понимаются четыре вещи: расширенные возможности программирования, расширенные возможности коммуникаций, расширенные интерфейсные возможности и расширенные возможности хранения данных; фактически три последние непосредственно вытекают из первой.

Аппаратной базой для виртуальных ПЛК в DACHS являются IBM PC совместимые ЭВМ: поскольку благодаря своим расширенным возможностям виртуальные ПЛК в DACHS могут выполнять самые разнообразные функции, конструктивное исполнение их может быть самым разнообразным. Например, при использовании виртуального ПЛК в качестве низового контроллера в зависимости от требуемой компоновки и разрядности системной шины (8 или 16 бит) это может быть либо PC/104 или MicroPC (Advantech, Octagon Systems, Fastwel), либо AT96 (Lippert) и т.п. Практически все современные встраиваемые контроллеры оборудованы твердотельными дисками CompactFlash, DiskOnChip и т.п., что облегчает разработку и обслуживание программного обеспечения.

При использовании виртуального ПЛК в качестве цехового контроллера удобным решением является реализация его на базе промышленной рабочей станции или панельного ПК (богатый выбор таких решений есть у компании Advantech) с необходимой периферией и возможностью эксплуатации в жёстких промышленных условиях.

Расширенные возможности программирования

Оговоримся сразу, что речь идет не о непосредственном расширении возможностей программирования логики виртуальных ПЛК, а о расширенных возможностях ее дополнения в рамках все того же МЭК 61131-3, причем в основном дело касается языка функциональных блоков.

Классический вариант расширения ПЛК новыми функциями подразумевает написание модулей расширения на C и оформление их в библиотеку функциональных блоков для среды программирования МЭК 61131-3. Разумеется, эти модули должны поддерживаться виртуальной машиной МЭК 61131-3, для этого в нее тоже необходимо внести соответствующие изменения. Как это сделать? Классический способ расширения виртуальной машины МЭК

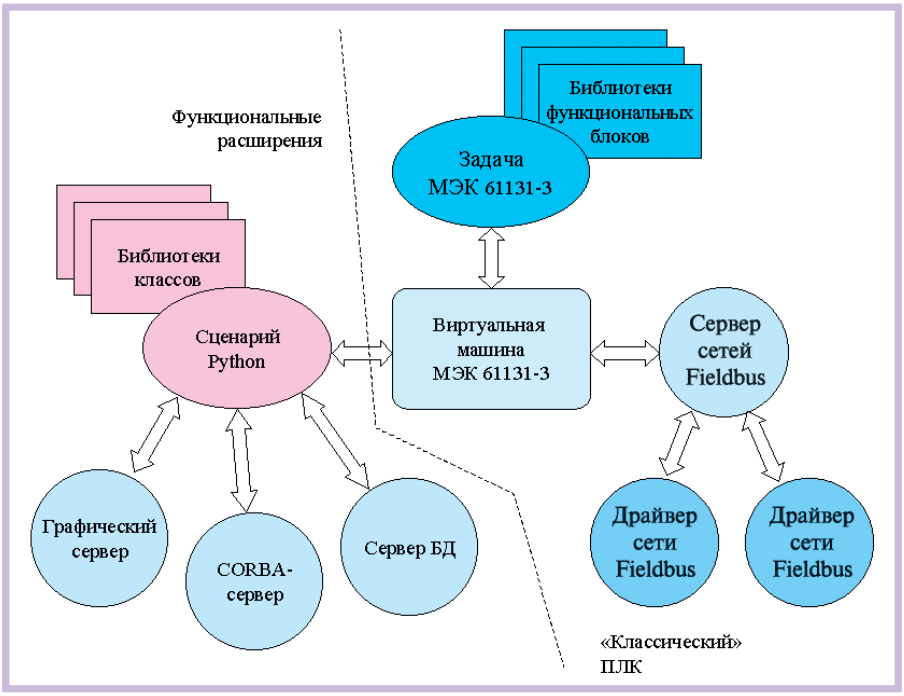

Рис. 1. Расширенный виртуальный ПЛК в DACHS®

61131-3 заключается в дописывании дополнительного кода на C. Этот метод содержит несколько неприятных подводных камней: во-первых, «распухает» сама целевая задача, а во-вторых, любое изменение кода виртуальной машины несет опасность внесения в него ошибки, а значит, угрожает задачам МЭК 61131-3, выполняющимся под ее управлением. Альтернативным способом расширения виртуальной машины МЭК 61131-3 может служить использование динамических библиотек (DLL), однако, хотя этот подход и позволяет сэкономить на объёме кода виртуальной машины, безопасности программному обеспечению не добавляет.

Разработчики виртуальной машины МЭК 61131-3 для DACHS пошли по другому пути (рис. 1) Как виртуальная машина, так и ее модули расширения в виртуальных ПЛК DACHS оформлены как отдельные процессы в операционной системе; модули расширения можно запускать и выгружать динамически — на объем кода виртуальной машины они не влияют. Мало того, являясь отдельными процессами, они выполняются каждый в своем собственном защищенном адресном пространстве и не могут навредить ни друг другу, ни виртуальной машине. Но и это еще не все. Вариативную (то есть зависящую от конкретного приложения) часть модуля расширения тоже можно изменять динамически – сделано это за счет применения в модулях расширения интерпретатора объектно-ориентированного языка сценариев Python; эта концепция получила название PyDACHS. Инвариантная часть модуля расширения оформляется как подключаемая библиотека классов Python, вариативная же пишется как сценарий Python, и ее можно в любой момент отредактировать обычным текстовым редактором и импортировать снова, что, кстати, еще и упрощает обслуживание на стадии тестирования.

Почему был выбран именно Python? Можно назвать как минимум пять причин. Во-первых, интерпретация дает гораздо более короткий цикл отладки: результат корректировки исходного текста можно проверить сразу, не прибегая каждый раз к цепочке «компиляция — компоновка — прогон». Во-вторых, объектная ориентация упрощает проектирование, особенно если говорить о сопряжении со средствами программирования МЭК 61131-3: объекты Python — фактически готовые функциональные блоки. В-третьих, код Python компактен: выигрыш по сравнению с C++ может достигать 5-10 раз! В-четвертых, Python очень популярен, хорошо документирован, и для него существует огромный выбор прикладного программного обеспечения с открытым исходным текстом. И, наконец, в-пятых, Python легко расширяем — дополнительные библиотеки классов без труда реализуются на C или C++.

Изменения, которые при этом претерпела сама виртуальная машина МЭК 61131-3, оказались минимальны: в нее просто добавились примитивы межзадачного взаимодействия (IPC – Inter-Process Communication), позволяющие задачам МЭК 61131-3 стандартным способом обращаться к модулям расширения. Таким образом, с точки зрения задач МЭК 61131-3, исполняющую их виртуальную машину можно рассматривать как микроядро с динамически подключаемыми сервисами.

К операционной системе (ОС), под управлением которой работает такой виртуальный ПЛК, должны предъявляться достаточно жесткие требования. Во-первых, система управления должна быть предсказуемой – потеря данных в системе управления абсолютно недопустима. Следовательно, применяется ОС жесткого реального времени с вытесняющей многозадачностью. Во-вторых, необходимо поддерживать большое количество различного оборудования: коммуникационных адаптеров, плат ввода-вывода и т.п., — ассортимент которого постоянно меняется. Значит, применяемая ОС должна предоставлять простые механизмы интерфейса с оборудованием, чтобы можно было в случае необходимости быстро разработать нужный драйвер. В-третьих, ОС должна легко встраиваться в устройства с ограниченными ресурсами и поддерживать бездисковые конфигурации. И, наконец, в-четвертых, для повышения отказоустойчивости необходимо обеспечить безопасность процессов, чтобы они своими действиями не могли навредить как друг другу, так и системе. Это реализуется в ОС, которые предоставляют процессам отдельные защищенные адресные пространства.

Исходя из перечисленных критериев, для реализации виртуального ПЛК в DACHS была выбрана встраиваемая ОС жесткого реального времени QNX. Сначала это была QNX4, теперь же в DACHS применяется и QNX6, что позволило еще больше понизить «порог встраиваемости». Дополнительным преимуществом QNX является то, что она обладает встроенной поддержкой прозрачных механизмов сетевого взаимодействия, позволяя объединять узлы QNX-сети в единую логическую совокупность ресурсов и упрощая тем самым построение распределенных систем. В ряде отраслей очень популярны SCADA-системы, работающие под управлением QNX, соответственно применение в таких случаях QNX и на

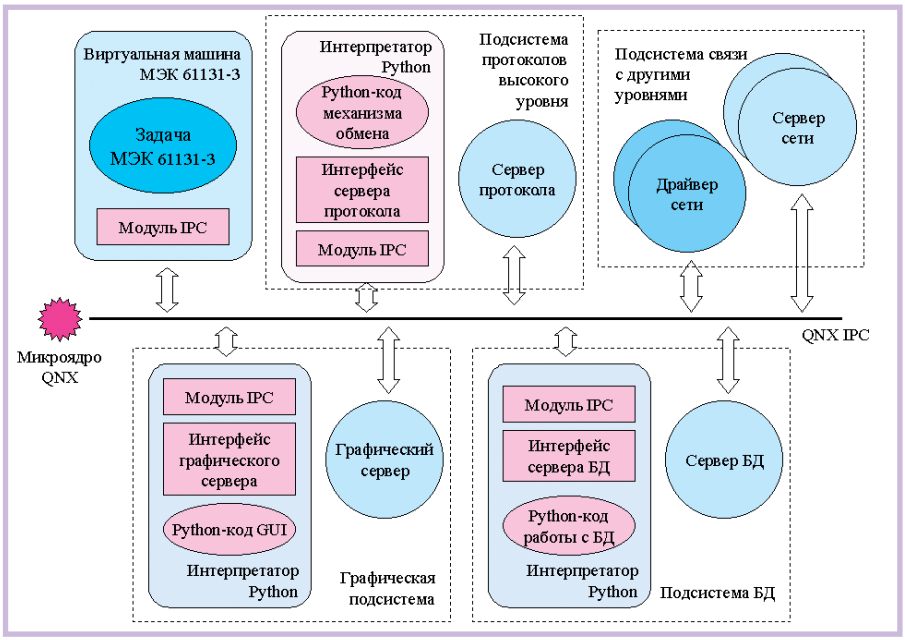

Рис. 2. Программная реализация расширенного виртуального ПЛК в DACHS®

уровне низовых контроллеров значительно упрощает интеграцию уровней.

Схема программной организации виртуального ПЛК в DACHS приведена на рис. 2. В данный пример включены, кроме собственно виртуальной машины МЭК 61131-3, два сервера сетевого взаимодействия (они предоставляют процессам сетевые сервисы) и два соответствующих драйвера сети, а также три модуля расширения на основе интерпретатора Python: подсистема протоколов высокого уровня, графическая подсистема и подсистема БД. Все компоненты общаются друг с другом посредством программной шины QNX IPC — механизма межзадачного взаимодействия QNX на основе обмена сообщениями. Для этого в те из них, которые изначально не поддерживали этот механизм, добавлен модуль QNX IPC — для виртуальной машины МЭК 61131-3 это означало расширение ее собственного программного кода, для интерпретатора же Python этот модуль подключается как внешняя библиотека классов. Казалось бы, без внесения изменений в код виртуальной машины обойтись все-таки не удалось, а значит, где обещанный выигрыш по надежности? Выигрыш по надежности здесь состоит в том, что модуль IPC — единственное изменение, внесенное в код виртуальной машины. Механизм обмена сообщениями универсален, а значит, будучи однажды реализован, он обеспечивает поддерживающим его задачам доступ сразу ко всем имеющимся сервисам. В случае же «классического» варианта пришлось бы всякий раз дописывать код виртуальной машины с добавлением каждого нового модуля расширения.

Работают модули расширения предельно просто. Запрашивая определенный сервис, задача МЭК 61131-3 фактически через встроенный в виртуальную машину модуль IPC вызывает соответствующий сценарий Python, ко-

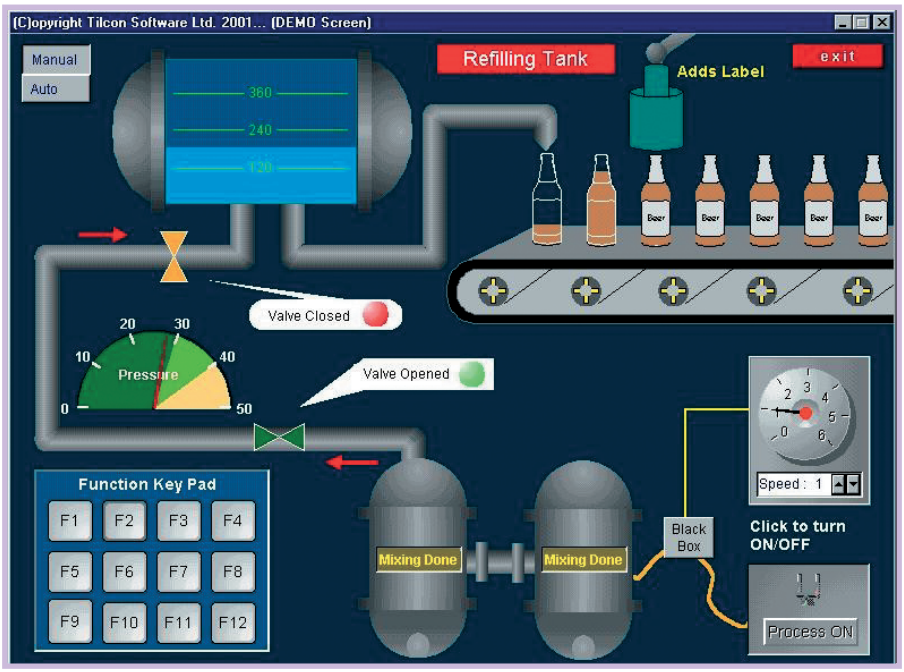

Рис. 3. Пример графического интерфейса, реализованного с помощью Tilcon

торый и отрабатывает запрос, обращаясь по мере необходимости к нужному серверу (посредством QNX IPC) через подключенную интерфейсную библиотеку. Если возникает необходимость изменить схему отработки того или иного запроса, достаточно внести коррективы в соответствующий сценарий Python, поскольку именно он несет специфичную для приложения смысловую нагрузку. Библиотека интерфейса с сервером и библиотека IPC при этом остаются теми же самыми.

На настоящий момент разработано множество интерфейсных библиотек PyDACHS, позволяющих писать сценарии Python для работы как с серверами БД, так и с графическими и коммуникационными серверами.

Расширенные возможности коммуникаций

Одна из слабых сторон ПЛК — их ограниченные способности коммуникаций как на уровне стыков, так и на уровне протоколов. Возможно, именно это как раз и привело разработчиков к мысли использовать PC-архитектуру в системах управления. Но если до сих пор ситуация была в известной мере паритетной из-за подчинения как «реальных», так и виртуальных ПЛК ограничениям одного и того же стандарта, то теперь наметился явный перелом в пользу решений на основе виртуальных ПЛК, и вот почему.

В 2001 году наконец вышел проект стандарта МЭК 61131-5, описывающий средства коммуникаций для ПЛК. И тут встал вопрос: а как реализовать предусмотренные этим стандартом расширения в среде исполнения МЭК 61131-3? У специалистов компании Steinhoff, как видно из изложенного, ответ был готов, что позволило им без труда за короткий срок реализовать дополнительный модуль расширения, обеспечивающий поддержку самых разнообразных механизмов коммуникации.

Виртуальные ПЛК в DACHS поддерживают большой выбор стыков и протоколов, включая протоколы высокого уровня из области Интернет-технологий, что обеспечивает этим контроллерам высокий уровень совместимости. В частности, для сопряжения с устройствами нижнего уровня АСУ ТП поддерживаются разнообразные промышленные сети (Profibus, CAN, LON, ASI, Interbus), сопряжение же с верхними уровнями возможно практически по любому стыку с использованием как QNX-сети, так и TCP/IP (в том числе и FTP, HTTP, ODBC, SNMP и т. п.) — работа с соответствующими серверами поддерживается посредством PyDACHS.

Кроме того, возможна интеграция виртуальных ПЛК DACHS со SCADA-системами, работающими под управлением Windows, посредством OPC-сервера, поставляемого в комплекте со средой программирования МЭК 61131-3.

Коммуникационное аппаратное обеспечение, поддерживаемое в DACHS, насчитывает десятки производителей, в число которых входят Allen Bradley, Echelon, Phoenix Contact, Siemens, Softing, SST и многие другие.

Открытые стандарты коммуникаций позволяют использовать в решениях на базе DACHS коммуникационную аппаратуру и распределенные УСО от самых различных производителей (Siemens, Entrelec-Schiele, и т.д.), но особого внимания здесь заслуживает модульная система распределенного ввода-вывода Wago I/O фирмы Wago. В ней для передачи данных могут использоваться до семи различных вариантов промышленных сетей, причем для перехода с одного типа сети на другой достаточно просто заменить соответствующий интерфейсный модуль в узле системы,— конфигурация модулей ввода-вывода останется прежней. Полезная хитрость использования такой системы в DACHS заключается в том, что, с точки зрения задачи 61131-3, все системы ввода-вывода одинаковы, поскольку сервисы, предоставляемые виртуальной машине сетевым сервером (в данном случае — сервером полевой шины), унифицированы и от типа применяемой сети не зависят. Таким образом, если будет необходимо перейти на другую промышленную сеть (например, система была изначально разработана для CAN, а потом понадобилось добавить поддержку Profibus-DP), достаточно будет сменить интерфейсный сетевой модуль станции Wago I/O, поставить в виртуальный ПЛК другую плату расширения и запустить соответствующий драйвер. Ни система ввода-вывода, ни задача МЭК 61131-3 при этом ни в какой модернизации нуждаться не будут.

Расширенные интерфейсные возможности

Нужен или нет виртуальному ПЛК графический интерфейс пользователя – вопрос спорный. В первую очередь это зависит от того, где и в какой роли будет применяться этот виртуальный ПЛК. С другой стороны, коль скоро виртуальные ПЛК часто применяются там, где интерфейс пользователя нужен — хотя бы в тех же цеховых контроллерах — предусмотреть такую возможность следует.

Модуль расширения, реализующий графическую подсистему виртуального ПЛК в DACHS, также реализован при помощи PyDACHS. Роль графического сервера при этом выполняет кросс-платформенный графический сервер Tilcon (продукт канадской компании Tilcon Software).

Сервер Tilcon предоставляет богатый выбор графических примитивов, поз-

воляя реализовывать самые разнообразные объекты, включая сложные области анимации (рис. 3). Еще одной отличительной особенностью Tilcon является наличие встроенного TCP/IP-сервера, позволяющего реализовывать технологию удаленного пользовательского интерфейса в гетерогенных сетях. Иными словами, оборудованный такой графической подсистемой виртуальный ПЛК может иметь не только локальный, но и распределенный по сети графический интерфейс, причем отображение и взаимодействие с оператором может вестись на удаленной рабочей станции под управлением любой из поддерживаемых сервером Tilcon ОС. Это предоставляет дополнительные возможности интеграции.

Расширенные возможности хранения данных

Ещё одно семейство модулей расширения PyDACHS дополняет виртуальные ПЛК способностью работать с данными при помощи СУБД. Подчеркнем особо: не складывать данные в БД, расположенную на верхнем уровне, а именно обеспечивать возможность локального хранения данных. И важно это в том числе из-за того, что когда данных становится много, встает вопрос автоматизации их обработки. Недостаток на рынке СУБД-приложений, поддерживаемых применяемыми на нижних уровнях встраиваемыми ОС реального времени, привел к тому, что традиционным решением стало использование СУБД только на верхних уровнях. Применение же СУБД на нижних уровнях позволяет не только облегчить жизнь программистам, но и восстановить естественный порядок вещей.

На текущий момент библиотеками PyDACHS поддерживаются такие СУБД как Berkeley DB и GDBM.

Что из этого следует?

Использование открытых стандартов при построении системы автоматизации позволяет смотреть на несколько шагов вперед и решать проблемы интеграции еще до их реального возникновения. Самые удачные в «локальном» смысле закрытые решения автоматизации часто не учитывают перспектив интеграции с пограничными областями, что приводит к буквально анекдотическим случаям. В частности, известен пример, когда связующим звеном между подсистемами АСУ ТП и АСУП являлся оператор, считывавший данные с экрана цехового контроллера и вручную заносивший их в корпоративную базу данных. Смех, да и только.

Заложенная в DACHS открытая модульная концепция позволяет набирать систему автоматизации — как ее аппаратную, так и программную стороны — по частям, используя только те «кирпичики», которые действительно необходимы, и связывая их между собой так, как этого требуют установленные ограничения. Один из возможных вариантов конфигурации системы управления на базе DACHS приведен на рис. 4.

В данной конфигурации используются интеллектуальные устройства нижнего уровня, подключенные к виртуальным ПЛК (здесь они выступают в роли низовых контроллеров) при помощи промышленной сети, низовые контроллеры объединены с цеховым контроллером по сегменту промышленной сети или Ethernet.

Обратите особое внимание на роль цехового контроллера: кроме выполнения своих основных задач, он служит как бы информационным шлюзом между подсистемой АСУ ТП и сетью предприятия. Это как раз один из спо-

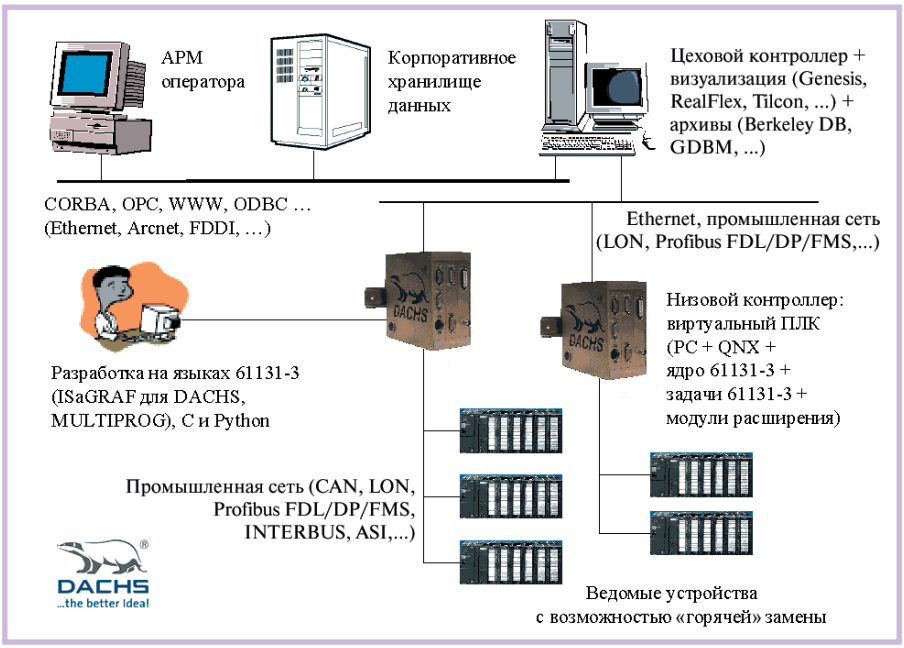

Рис. 4. Пример конфигурации системы на базе DACHS®

собов избежать «вавилонской башни» при организации взаимодействия между информационными подсистемами, когда у каждой из них существуют определенные традиции хранения и обработки информации (и, разумеется, разные, поскольку сам характер информации в них отличается), а необходимость сопряжения очевидна. В нашем примере проблема решается путем применения в цеховом контроллере технологической СУБД с открытым протоколом доступа к данным, «понятным» как для контроллеров нижних уровней, записывающих данные в архив, так и для ЭВМ сети предприятия, использующих эти данные на верхних уровнях.

Заключение

Так где же зарыт барсук? Решения на базе DACHS широко используются по всему миру в самых разнообразных отраслях (промышленная автоматизация, транспорт, энергетика, робототехника, почтовые системы, медицинская техника, автоматизация зданий и т.д.). Под управлением DACHS работают системы управления дизельными генераторами электропитания на наземных станциях Европейского космического агентства Коуроу и Курина, нефтяные платформы компании Thule RigTech, расположенные в Шотландии, Норвегии, Франции и Сингапуре, метрополитен в Бухаресте и многие другие системы.

Будем надеяться, что заложенные в DACHS принципы совместимости помогут и российским инженерам строить красивые, функциональные и надежные системы управления, давая возможность не задумываться на каждом шагу о возможных проблемах интеграции решений между собой. ●

Автор — сотрудник SWD Software Ltd.
Телефон: (812) 443-0260
Факс: (812) 443-0497