OPC Unified Architecture: изменения в популярной технологии информационных обменов с точки зрения инженера

Николай Богданов, Ольга Киселёва

В статье рассмотрены основные принципы и особенности новой унифицированной архитектуры – OPC UA. Эта технология позволяет использовать преимущества универсального интерфейса для взаимодействия аппаратного и верхнего уровней автоматизации предприятий.

Введение

Для упрощения и удешевления реализации информационных обменов в системах промышленной автоматизации в 1996 году была предложена технология OPC (OLE for Process Control), в которой единообразие в настройках стыков SCADA-системы с «внешним миром» достигается за счёт использования определённого шлюза, унифицирующего интерфейс взаимодействия с клиентом и скрывающего частный протокол отдельных средств автоматизации. Использование «классической» OPC ограничено платформой Windows, так как это не протокол передачи данных, а именно программная технология, основанная на механизме удалённого вызова процедур с использованием стека DCOM. Это накладывает свой негативный отпечаток на такие параметры процесса взаимодействия по OPC, как безопасность, надёжность и резервирование. Проанализировав недочёты текущих спецификаций, ассоциация OPC Foundation разработала оригинальную унифицированную технологию, которая добавляет новые возможности, призванные помочь пользователям построить усовершенствованные решения для систем автоматизации.

Какие изменения в типах передачи данных, архитектуре доступа к различным категориям данных и в структуре программного обеспечения несёт в себе новое поколение OPC? Какие преимущества даёт эта технология? На эти вопросы мы постараемся ответить в статье, используя разработки одного из первых и наиболее последовательных участников OPC Foundation – компании ICONICS, которая в этом году добавила поддержку OPC Unified Architecture (OPC UA) в свои ключевые продукты.

OPC Unified Architecture

Типовая схема использования OPC для доступа к данным ПЛК и SCADA-систем показана на рис. 1.

Говоря о «классической» OPC, мы в первую очередь имеем в виду передачу данных согласно спецификациям OPC DA (Data Access – в масштабе реального времени), OPC HDA (Historical Data Access – архивов изменений параметров) и OPC A&E (Alarm and Events – тревог и событий). Популярность последних двух спецификаций существенно меньше, чем у OPC DA, не в последнюю очередь потому, что передача данных архивов и аварийных событий требовала от производителя оборудования разработки ещё двух отдельных программ, а от разработчика системы диспетчеризации – настройки ещё одного или двух дополнительных информационных стыков с серверами OPC HDA и OPC A&E, имеющими независимые и не связанные с OPC DA адресные пространства. В OPC UA предусматривается объединение механизмов адресации и доступа к разным категориям данных.

Итак, *первая особенность OPC UA* – получение текущих и архивных значений параметров и событий происходит теперь единообразно от одного источника. При этом унифицированное адресное пространство ещё и содержит больше семантических сведений, доступных при его просмотре (browsing). Для примера рассмотрим произвольный объект «бойлер», которым мы должны управлять через события «включён/выключен», по изменению

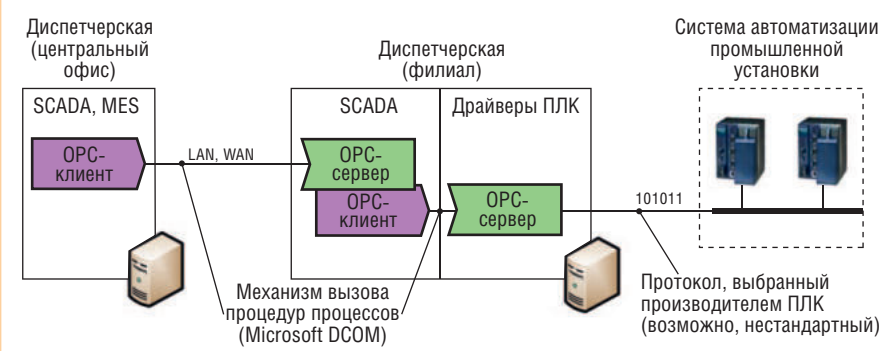

Рис. 1. Схема применения «классической» OPC

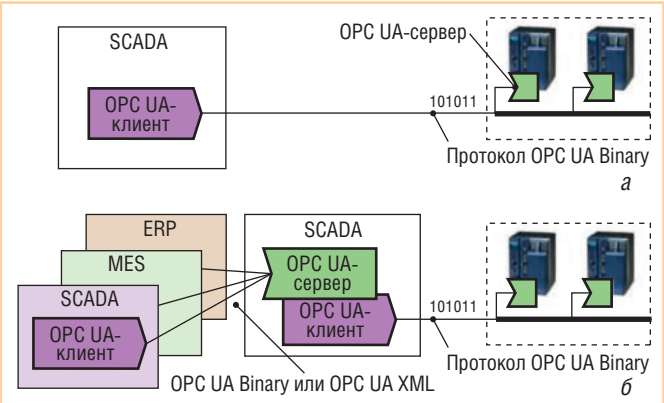

Рис. 2. Возможные схемы взаимодействия с OPC-серверами в ПЛК

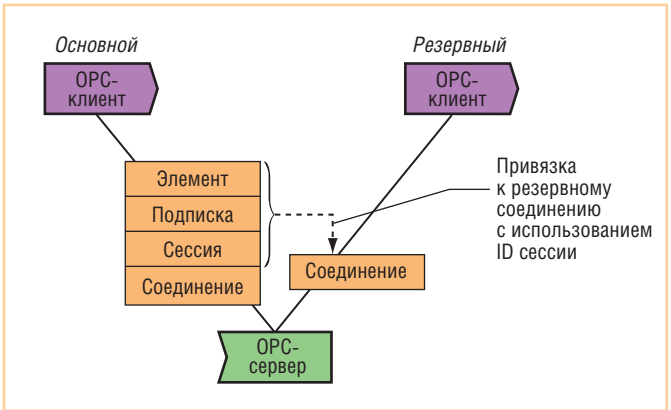

Рис. 3. Сущность взаимодействия OPC-клиента и OPC-сервера

температуры и давления, а также анализировать, как эти параметры влияют друг на друга. Логично, что данные свойства нужно группировать и анализировать все вместе. Семантика OPC UA позволяет это сделать. Один объект здесь представлен набором свойств (температура, давление), методов (включён/выключен) и событий (температура слишком высокая, давление слишком низкое).

Вторая особенность OPC UA – полностью переработанный механизм взаимодействия OPC-клиента и OPC-сервера. Произошёл отказ от DCOM в пользу обмена бинарными или XML-сообщениями (то есть OPC UA – это именно протокол передачи данных; к такому механизму намного «дружественнее» относятся межсетевые экраны – firewall – и прочие средства информационной безопасности; с другой стороны, отказ от DCOM вызван и ростом популярности решений под Linux в 2000-х гг.). Также стандарт определяет гибкий механизм управления сетевыми и логическими соединениями OPC-серверов и OPC-клиентов для поддержки резервированных конфигураций и т.п. Более того, интерфейс OPC-сервера рассматривается как набор сервисов, а значит, данные систем автоматизации могут стать доступными другим приложениям в единой сервисной шине предприятия (ESB), построенной по технологии SOA.

Третья особенность OPC UA – отказ от использования механизмов контроля прав доступа Windows (основанных на проверке имени/пароля пользователя, от имени которого запускается OPC-клиент), разработчики OPC UA предложили более современный способ контроля с использованием сертификатов. Также предусмотрена возможность шифрования передаваемых данных.

Естественно, уделено внимание и вопросу сохранения уже сделанных предприятиями инвестиций. Использование «классической» OPC возможно и в OPC UA-среде: специальная оболочка (wrapper) обеспечивает доступ к обычному OPC DA-серверу, а proxy-модуль позволяет OPC DA-клиенту взаимодействовать с новыми OPC UA-серверами.

Рассмотрим подробнее технические особенности OPC Unified Architecture и возможность их использования для изменения существующего порядка разработки систем диспетчерского управления.

Структура программного обеспечения

Переход от межпрограммного взаимодействия поверх DCOM к передаче пакетов позволяет исполнять OPC-серверы и OPC-клиенты не только на компьютерах под управлением отличных от Windows операционных систем, но и в самих контроллерах (рис. 2*а*). Аргументом в пользу сохранения существующей схемы, когда OPC-сервер устанавливается на компьютере, может служить ограниченность ресурсов и, в частности, оперативной памяти контроллера, что не позволит ему одновременно обслуживать нескольких подключённых клиентов. Однако множество «внешних» клиентов может взаимодействовать со SCADA-системой (как показано на рис. 2*б*). При любом варианте отказ от использования проприетарного протокола в пользу OPC UA для взаимодействия с ПЛК – это сокращение разнотипности средств, используемых при построении системы автоматизации, приводящее не только к снижению трудозатрат при пусконаладочных работах, но и облегчающее модернизацию отдельных компонентов системы в будущем. Также при реализации сценария удалённого взаимодействия OPC-клиента и OPC-сервера (не через ЛВС, а через региональную сеть передачи данных в масштабе региона – WAN), когда трафик TCP/IP проходит через маршрутизаторы и межсетевые экраны (см. взаимодействие центрального офиса и филиала на рис. 1), «классическая» OPC требует использования программного обеспечения промежуточного слоя, например, ICONICS GenBroker или ICONICS DataWorX OPC Tunnelling. OPC UA не использует такой прослойки, ей безразлично, установлены ли клиент и сервер на компьютерах, находящихся в соседних комнатах или в разных городах.

Быстрое реагирование на сбои и резервирование передачи данных

OPC UA разделяет несколько уровней взаимодействия OPC-клиентов и OPC-сервера. Во-первых, каждый из клиентов устанавливают с сервером своё защищённое сетевое соединение. При этом если в «классической» OPC право доступа клиента к серверу определялось, исходя из прав пользователей Windows, от чьего имени они запускались на соответствующих компьютерах, то в OPC UA клиент и сервер идентифицируют себя цифровыми сертификатами. Во-вторых, в рамках соединения создаётся сессия – логическое соединение клиента и сервера. Параметром сессии являются уже права отдельного пользователя, использующего OPC-клиент, так как OPC-сервер может вводить ограничения на операции чтения/записи отдельных элементов для разных пользователей. Уже в рамках сессии производится собственно передача данных (выполнение запросов на чтение/запись), а также производится инициализация списка элементов, об изменениях значений которых сервер направляет клиенту уведомление (рис. 3 – между соединением, сессией,

подпиской, элементом отношения «один ко многим»). Если сбой в канале передачи данных приводит к разрыву сетевого соединения, то после установления нового соединения созданную ранее сессию можно «привязать» к нему и продолжить работу без повторной инициализации, то есть обеспечивается возможность быстрого восстановления передачи данных. Аналогично при реализации сценария резервирования: если есть основной и резервный OPC-клиенты, ведущие опрос одного OPC-сервера, то соединение с сервером устанавливают оба, а создаёт сессию и ведет опрос основной OPC-клиент. В случае его краха резервный OPC-клиент подключает сохранённую на сервере сессию к своему соединению и продолжает получение данных.

Отдельно отметим изменения в концепции отправки OPC-сервером уведомлений об изменениях значений отдельных параметров клиентам (механизме так называемого спонтанного уведомления об изменениях SRBX – spontaneous report by exception). В OPC DA этот механизм был реализован через обратный вызов методов клиента, в OPC UA межпрограммное взаимодействие не используется, а отправляются сообщения. Важной особенностью их отправки является то, что сервер сам определяет момент отправки (по факту изменения значений отдельных параметров), но не создаёт новых сообщений, а выбирает их из очереди ранее полученных «заготовок» от данного клиента. То есть сначала клиент направляет серверу массив запросов об изменениях (с уникальными ID), сервер их кэширует и по мере появления изменений (или при прохождении заданного периода времени без изменений) отправляет клиенту; далее клиент подтверждает факт получения сообщения. Периодически клиент пополняет очередь на сервере сообщениями с новыми ID. Таким образом исключается возможность неконтролируемого роста числа отправляемых сервером сообщений об изменениях.

В механизме передачи уведомлений также проявляется независимость программного соединения – в случае сбоя и последующего восстановления сетевого соединения сервер может передать клиенту весь подготовленный за время отсутствия связи массив сообщений об изменениях.

Транзакционная передача данных

Как уже отмечалось, OPC UA реализует важные изменения и в части организации данных сервера, то есть адресного пространства. Напомним, что в OPC DA теги были сгруппированы иерархически с помощью папок. У каждого тега были обязательные свойства: значение (скалярное или векторное, одного из предопределённых простых типов данных – Int16, Float, String и т.п.); признак достоверности (quality); метка времени. Разработчик OPC-сервера мог определять дополнительные свойства для тегов, такие как описание единиц измерения и т.п.; было желательным, чтобы OPC-клиент мог считывать значения этих дополнительных свойств наряду с обязательными свойствами. В OPC UA адресное пространство организовано иерархически за счёт введения объектов (в терминах объектно-ориентированного программирования), то есть экземпляров классов. Класс (и объект) может содержать переменные, ссылки на другие объекты и даже методы – функции, доступные для вызова OPC-клиентом. При этом тип переменной может быть сложным, аналогично понятию структуры из языка программирования, объединяя поля разных типов, включая таблицы (массивы) и т.д.

В частности, это даёт возможность отойти от традиционной для систем контроля и управления передачи значений отдельных сигналов и обеспечить передачу наборов данных – таблиц, фиксирующих значения целого ряда параметров на выбранный момент времени, что более приемлемо для MES- и ERP-систем. Можно предположить, что формирование этих сложно структурированных переменных будет также происходить не автоматически, а по команде, то есть при вызове специального метода в OPC-сервере клиентом.

Адаптивность задачи информационного стыка с OPC-сервером

В адресном пространстве OPC UA-сервера может быть реализована явная типизация объектов, то есть сопоставление однотипных групп технологических параметров шаблону (классу). Благодаря этому появляется возможность в приложении-клиенте контролировать полноту получаемой информации и в случае изменения её объёмов в сервере автоматически на это реагировать.

OPC UA-клиент может отойти от принятой в настоящее время схемы, когда перечень сигналов обмена с OPC-сервером определяется в виде списка имён тегов. Явное предоставление семантической информации в OPC UA-сервере и, в частности, наличие ссылок от описания класса к его экземплярам позволяет задать шаблон перечня объектов, а не вводить на клиенте список имён экземпляров класса, существующих на момент настройки стыка. OPC UA-клиент также может подписаться на получение уведомлений от сервера об изменениях в адресном пространстве. И, например, в случае систем, формирующих на выходе сводные отчёты, возможность автоматически отреагировать на увеличение или уменьшение количества анализируемых объектов, несомненно, полезна.

На рис. 4 показан пример: в адресном пространстве OPC UA-сервера может быть задано определение класса со списком параметров, поступающих от систем автоматизации и характеризующих режим работы скважины на подземном хранилище или месторождении газа, нефти. OPC-клиент, формирующий отчёт по добыче за предыдущий час, может получить список обратных ссылок типа *HasTypeDefinition,* указывающих от объектов на их класс, то есть получить массив имен объектов и составить запрос на чтение значений переменных *QLH* из каждого объекта. При обнаружении изменения в списке ссылок клиент обновляет свой перечень имён и идентификаторов объектов (то есть проводит повторную частичную инициализацию без прерывания соединения, если говорить на техническом

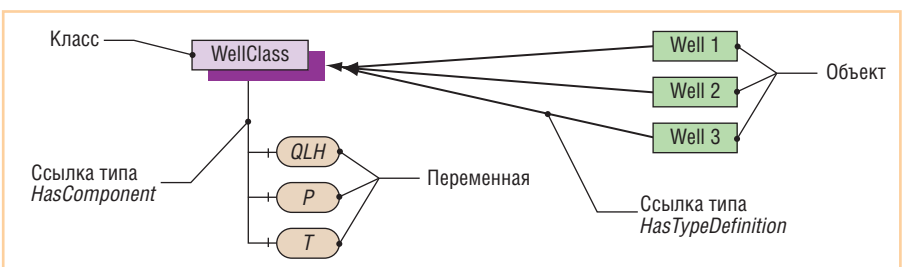

Рис. 4. Ссылки в адресном пространстве OPC UA-сервера

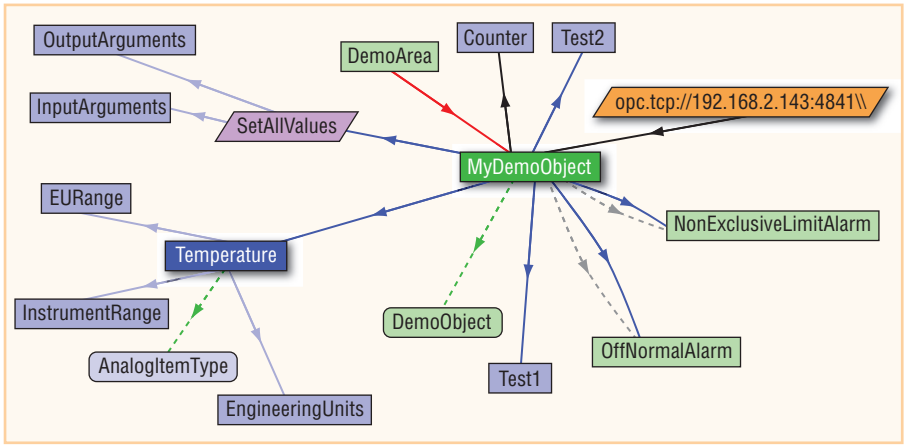

Рис. 5. Просмотр адресного пространства OPC UA-сервера

языке). Однако участие человека требуется на этапе начальной настройки стыка, так как отличить смысловую нагрузку значения, хранимого в переменной *QLH,* от значений переменных *P* или *T* можно, только прочитав атрибут – текстовое описание переменной – или проанализировав сами имена. Разработчики OPC UA рассматривают это как ограничение и предусматривают в будущем выпуск специализированных информационных моделей – документов в текстовом и XML-форматах, определяющих правила использования имён переменных для отдельных предметных областей.

Использование OPC UA в работе

Спецификация OPC UA, разработанная OPC Foundation, состоит на текущий момент из 13 частей и формальным образом описывает различные аспекты логики работы и взаимодействия OPC UA-серверов и клиентов. Текст спецификации распространяется только среди организаций-членов OPC Foundation, поэтому всем заинтересованным в более детальном изучении особенностей этого нового протокола мы рекомендуем первую из появившихся книг [1].

Однако сейчас, в 2010 году, поддержка OPC UA уже реализована в большом количестве коммерческих и бесплатных приложений. Продемонстрируем, как это работает, с использованием 64-битового пакета ICONICS GENESIS64 версии 10, демонстрационных UA-сервера и клиента разработки Unified Automation GmbH.

Запустим GraphWorX64. Создаём динамический элемент *Process Point,* выбираем источник данных – появляется окно Data Browser. Вводим непривычный адрес *opc.tcp://192.168.2.143:4841* и подключаемся к демо OPC UA-серверу. Обратите внимание: мы указали тип протокола, IP-адрес (можно было указать имя хоста), на котором запущен сервер, и номер порта. После подключения для просмотра доступно адресное пространство сервера, но традиционная древовидная структура неинформативна – используя табличное (Grid) или сетевое (Mesh) представления, можно увидеть, что элементы адресного пространства связаны ссылками различных типов. В частности, объект *MyDemoObject* (рис. 5) группирует (ссылка типа *HasComponent*) переменные *Temperature, Counter, Test1, Test2* и метод *SetAllValues.* К объектам-экземплярам классов тревог ведут также ссылки типа *HasCondition* и т.д.

Выберем переменную *Temperature* как источник данных для одного элемента *Process Point* и её свойство *EngineeringUnits* – для второго. Переведя GraphWorX64 в режим Runtime, увидим отображаемые значения (рис. 6; снимок

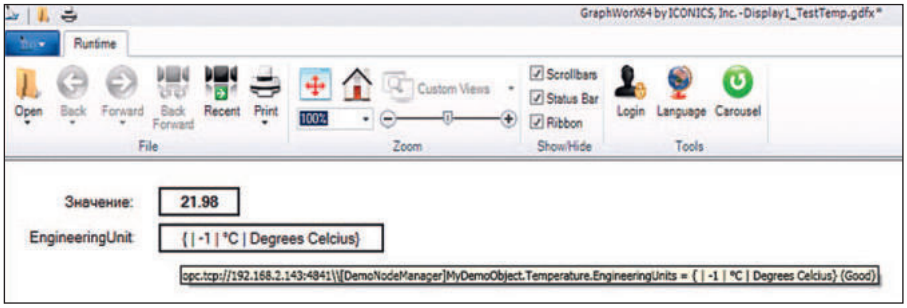

Рис. 6. Экранная форма GraphWorX64

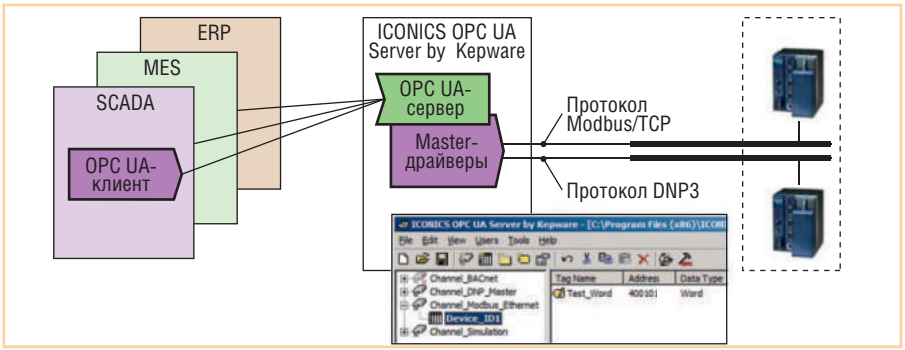

Рис. 7. Применение и конфигурация ICONICS OPC UA Server

экрана также содержит всплывающий при наведении мыши на второй из элементов *Process Point* поясняющий текст).

Данный пример, помимо очевидного подтверждения «это работает», демонстрирует упоминавшуюся ранее возможность представления в OPC UA-сервере и передаче клиенту не отдельных значений, а структур данных. В данном случае свойство *EngineeringUnits* содержит значение (*Value*), объединяющее элементы *UnitID, DisplayName, Description* (и еще, если быть точным, пустой *NameSpaceIndex*). Для адресации к конкретному элементу его имя нужно добавить к адресу (обратите внимание на разделитель «::»), например, чтобы отобразить только «°C», необходимо указать: *opc.tcp://192.168.2.143:4841 \\MyDemoObject.Temperature.EngineeringUnits::DisplayName*.

Возможности OPC UA-оболочек, то есть программного обеспечения доступа к «классическим» OPC-серверам с использованием OPC UA-клиентов, можно дополнительно испытать, загрузив, например, UA Wrapper компании Unified Automation. Если ранее подключение удалённого OPC-сервера требовало использования программного обеспечения туннелирования со стороны и клиента, и сервера (например, GenBroker из состава SCADA-пакета ICONICS GENESIS-32), то при применении современных SCADA-систем, имеющих OPC UA-клиента, можно ограничиться установкой дополнения в виде OPC UA-оболочки на стороне сервера. Эти продукты не имеют собственного графического интерфейса и служат только преобразователями протокола; при этом одновременно с их использованием сохраняется возможность и традиционного подключения к OPC-серверу.

С другой стороны, отказаться от отдельных традиционных OPC DA-серверов позволяет ICONICS OPC UA Server, разработанный ICONICS в партнерстве с фирмой Kepware. Этот продукт призван создать единую точку доступа к полевым устройствам автоматизации, так как он включает в себя драйверы для связи по более чем 100 протоколам (Allen-Bradley DH+, Omron FINS, Siemens S7 MPI, Triconex Ethernet, BACnet, DNP и многие другие). ICONICS OPC UA Server можно протестировать как отдельное приложение (в версиях Standard и Premium), а также в составе пакета GENESIS64 V10.

На рис. 7 показана схема применения этого OPC UA-сервера. Для каждого протокола, используемого на предприятии, вводится свой «канал», определяющий базовые характеристики стыка, затем для каждого отдельного устройства настраивается перечень сигналов. Эти перечни могут быть выгружены в CSV-файл, отредактированы в Excel и загружены обратно, что минимизирует трудозатраты инженера.

Заключение

Унифицированная архитектура OPC UA, ключевые особенности которой описаны в данной статье, позволяет расширить возможности по передаче информации между приложениями в пространстве промышленного предприятия. Объединяя стандартные OPC-спецификации, эта новая технология доработана с целью устранить текущие недостатки использования DCOM и способствовать повышению уровня интеграции в современные управляющие системы. ●

Литература

1. Mahnke W., Leitner S.-H., Damm M. OPC Unified Architecture. – Berlin: Springer, 2009. – ISBN 978-3-540-68898-3.

Авторы – сотрудники ЗАО «АтлантикТрансгазСистема» и ООО «ПРОСОФТ»
Телефон: (495) 234-0636
E-mail: info@prosoft.ru