УДК 681.518:658.512

АРХИТЕКТУРА ПРОГРАММНЫХ СЕРВИСОВ СИНХРОНИЗАЦИИ ДАННЫХ В РАСПРЕДЕЛЕННОЙ МНОГОУРОВНЕВОЙ WEB-SCADA СИСТЕМЕ

Н.В. Ткачук, канд. техн. наук

Национальный технический университет "Харьковский политехнический институт"

Разработана унифицированная программная архитектура подсистемы синхронизации данных в структуре сложной INTERNET - базированной информационно-управляющей системы класса АСУ ТП.

* * *

Розроблено уніфіковану програмну архітектуру підсистеми синхронізації даних у структурі складної INTERNET – базованої інформаційно-керуючої системи класу АСУ ТП.

* * *

An unified software architecture for data synchronization subsystem is developed, which is included in the framework of complex INTERNET-based process control system.

1. Актуальность проблемы синхронизации ресурсов данных в распределенных информационно- управляющих системах

При разработке многоуровневых распределенных информационно-управляющих систем (МРИУС) важная проблема их проектирования и сопровождения – синхронизация данных, которые накапливаются и обрабатываются в отдельных вычислительных узлах, расположенных на различных уровнях логической иерархии системы [1-3]. В настоящее время одним из новых типов МРИУС, которые активно разрабатываются и внедряются в различных отраслях промышленности, являются SCADA (*Supervisory Control And Data Acquisition*) системы [4-6], позволяющие вести сбор и обработку данных о сложных технологических процессах (ТП) в удаленном режиме. Для INTERNET-ориентированных SCADA систем (предлагаем в дальнейшем обозначать их как *Web-SCADA* системы) [7-9], эта проблема стоит особенно остро, т.к. они имеют открытую архитектуру и предполагают одновременную согласованную работу различных категорий пользователей в реальном масштабе времени, что невозможно без постоянного выполнения условий синхронизации данных. При этом характерно то, что такие системы, как правило, работают в реальном масштабе времени, в режиме «24x365», т.е. круглосуточно в течение ряда лет, и должны адекватно отображать телеметрические данные о сложных ТП [6,9].

Ниже рассмотрим программные решения для процедур синхронизации данных в сложных МРИУС, полученные нами в ходе разработки и реализации архитектуры региональной Web-базированной АСУ ТП для ряда объектов газопромыслового управления «Харьковгаздобыча» [8]. Географическая схема выполнения этих проектов в Харьковском регионе изображена на рис. 1.

2. Существующие подходы к решению задачи синхронизации данных в МРИУС и постановка задачи данного исследования

Концептуальный аспект процесса синхронизации данных поясняет рис. 2, на котором показаны уровни репликации данных в топологической структуре некоторой МРИУС. Предварительно следует уточнить несколько важных понятий, а именно [3]:

- *синхронизация данных* (*data synchronization*) - это процедура в распределенных системах баз данных (БД) с тиражированием,

Рис. 1. Фрагмент схемы региональной диспетчерской системы ГПУ „Харьковгаздобыча" (проект)

обеспечивающая отождествление дублирующих копий данных, или так называемых репликаций (*replication*), размещенных в различных узлах МРИУС;

- *тиражирование* или *репликация данных* (*data replication*) - это технология в распределенных БД, которая предусматривает поддержку копий некоторых фрагментов данных в нескольких узлах сети в целях приближения данных к месту их использования и сокращения за счет этого сетевого трафика и повышения производительности всей системы.

Узел сети, с которого происходит рассылка тиражируемых данных, иногда называют *издателем* (*publisher*), а узел, принимающий репликации данных, – соответственно, *подписчиком* (*subscriber*) [1]. Очевидно, что в структуре сложной иерархической МРИУС одни и те же узлы могут быть как издателями, так и подписчиками в процессе их синхронизации и каждый подписчик имеет, как правило, нескольких издателей (см. рис. 2).

Современные СУБД предоставляют различные решения для синхронизации данных в МРИУС. Их можно кратко рассмотреть на примере технологий репликации данных, реализованных в СУБД класса «клиент-сервер», к которой в полной мере относится, например, СУБД MS SQL Server 2000 [2].

Как правило, такая система может осуществлять три вида репликации:

1. *Репликация по «снимку»* (*snapshot replication*) пересылает данные в том виде, в котором они есть (полностью) на какой-либо момент времени и не отслеживает изменения в данных. Случаи, в которых целесообразно применять репликацию по снимку, следующие: данные меняются редко; постоянное обновление данных не является критическим; объемы пересылаемых данных малые.

   В нашем случае (см. ниже) репликация по снимку неприемлема из-за несоответствия всем трем требованиям, так как в Web-SCADA данные изменяются (добавляются) постоянно и объемы данных,

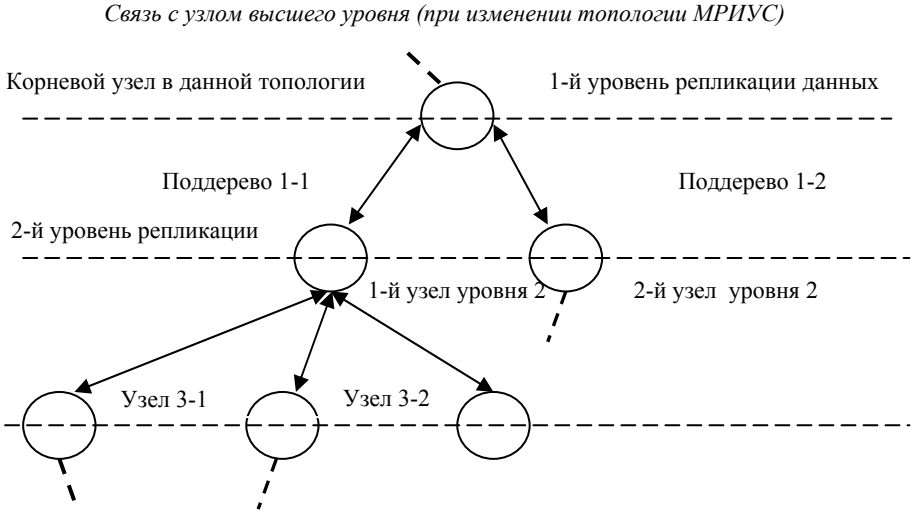

Рис. 2. Схема уровней репликации данных в структуре типовой МРИУС

пересылаемые при использовании алгоритма репликации по снимку, будут также постоянно возрастать, что, в конечном счете, приведет к невозможности осуществления синхронизации в приемлемые сроки, с учетом требования работы всей системы в реальном масштабе времени.

2. *Репликация-объединение* (*join-replication*) также требует снимка с БД, включающего ее схему и данные публикуемых таблиц. Снимок БД пересылается в требуемые узлы сети, после чего проводится синхронизация данных. Репликация-объединение эффективна в таких случаях: а) имеется множество подписчиков, изменяющих данные асинхронно; б) подписчики должны получать данные, изменять их, а затем отсылать назад издателю; в) не возникает множества смысловых конфликтов в синхронизируемых данных.

Применение этого метода репликации привело бы к чрезмерным затратам аппаратных ресурсов, так как в нашем случае имеется несколько издателей на одного подписчика и подписчики не изменяют получаемых данных. Кроме того, нет необходимости в пересылке данных издателю в обратном направлении.

3. *Транзакционная репликация* (*transaction replication*) является одним из самых эффективных методов репликации. При использовании этого метода выполняется начальный снимок данных в БД, после чего отслеживается журнал ее транзакций и проводится обновление данных на основании этого журнала. Модификации данных хранятся в специальной дистрибутивной БД, и на их основании проводится репликация с подписчиком.

На основе этих подходов нами было принято решение – реализовать некоторую комбинацию существующих алгоритмов с учетом необходимости удовлетворения следующим требованиям:

- подсистема синхронизации данных (ПСД) в реальных условиях должна реализовывать пересылку данных через модемное соединение, что всегда связано с неустойчивостью канала связи и вероятностью обрыва связи;
- для возможности масштабирования всей Web-SCADA системы, а также для повышения ее гибкости необходимо обеспечить независимость работы ПСД от структуры и

семантики пересылаемых данных, т.е. при их изменении логика работы ПСД не должна измениться;

- связь ПСД с интегрированной базой данных узла (ИБДУ) или *Integrated Node Database (INDB)* - о назначении и функциональности этого компонента всей Web-SCADA системы более подробно см. в [8,9] - должна обеспечиваться с помощью заранее определенного интерфейса, что также вызвано требованием гибкости и масштабируемости всей системы в целом;
- следует максимально автоматизировать работу ПСД для того, чтобы не ставить обслуживающий персонал отдельных объектов АСУ ТП перед необходимостью прохождения каких-либо специальных курсов, обучающих работе с ПСД, что всегда связано с дополнительными, и весьма существенными, материальными и временными затратами.

3. Архитектура разработанной подсистемы синхронизации данных (ПСД)

С учетом описанных выше требований, было принято решение реализовать архитектуру ПСД в виде двух наборов компонентов: клиентской и серверной. Они могут устанавливаться на каждом узле МРИУС, в результате чего этот узел получает возможность пересылать и принимать данные, т.е. может быть как издателем, так и подписчиком. Чтобы уменьшить участие оператора системы в работе ПСД, оба эти компонента должны быть реализованы в виде кода, выполняющегося в фоновом режиме, а именно: MS Windows NT / 2000 Service в качестве клиентской части и MS ASP (*Active Server Pages*) скрипт в качестве серверной части [10] .

Синхронизация проводится следующим образом: каждый узел, кроме узлов – «листьев» (т.е. у которых нет потомков), опрашивает своих прямых потомков на наличие у них новых данных. Если таковые имеются для синхронизации, то начинается их пересылка. Если этот узел, в свою очередь, является потомком какого-либо узла, то по запросу с его узла-предка выдаются все имеющиеся на данный момент на нем новые данные.

Разработанная программная архитектура включает в себя следующие компоненты, агрегирующие отдельные функции ПСД:

(1) *Thread Manager (ThrdMngr)* – этот компонент представляет собой COM-объект [8], реализующий функции менеджера вычислительных потоков в процедурах синхронизации; (2) *Dial Up Connection Manager (ConnMngr)* – компонент, отвечающий за модемное соединение с сервером ПСД, он также реализован в виде COM - объекта, использующего встроенную функциональность сервиса *Remote Access Service* в MS Windows [10]; (3) *Replication Client (ReplClnt)* - это компонент, реализующий клиентскую бизнес-логику ПСД с помощью MS XML Parser [10]; (4) *Replication Server* (*ReplSrv*) - серверная часть всей ПСД, которая отвечает за предоставление клиенту данных синхронизации (параметров ТП, системных сообщений, различных уставок и т.д. [6]).

Диаграмма размещения этих компонентов в UML-нотации показана на рис.3. В реальной МРИУС компоненты (1)-(3) располагаются на клиентской стороне подсистемы и выполняют функции клиента синхронизации (подписчика), т.е. запрашивают данные, предоставляя серверу маркер последнего момента времени, за который в ИБДУ текущего узла содержатся данные.

Компонент (4) выполняет функции сервера репликации и предоставляет актуальные данные, которые соответствуют всем моментам времени, после заданного, вплоть до текущего сеанса синхронизации.

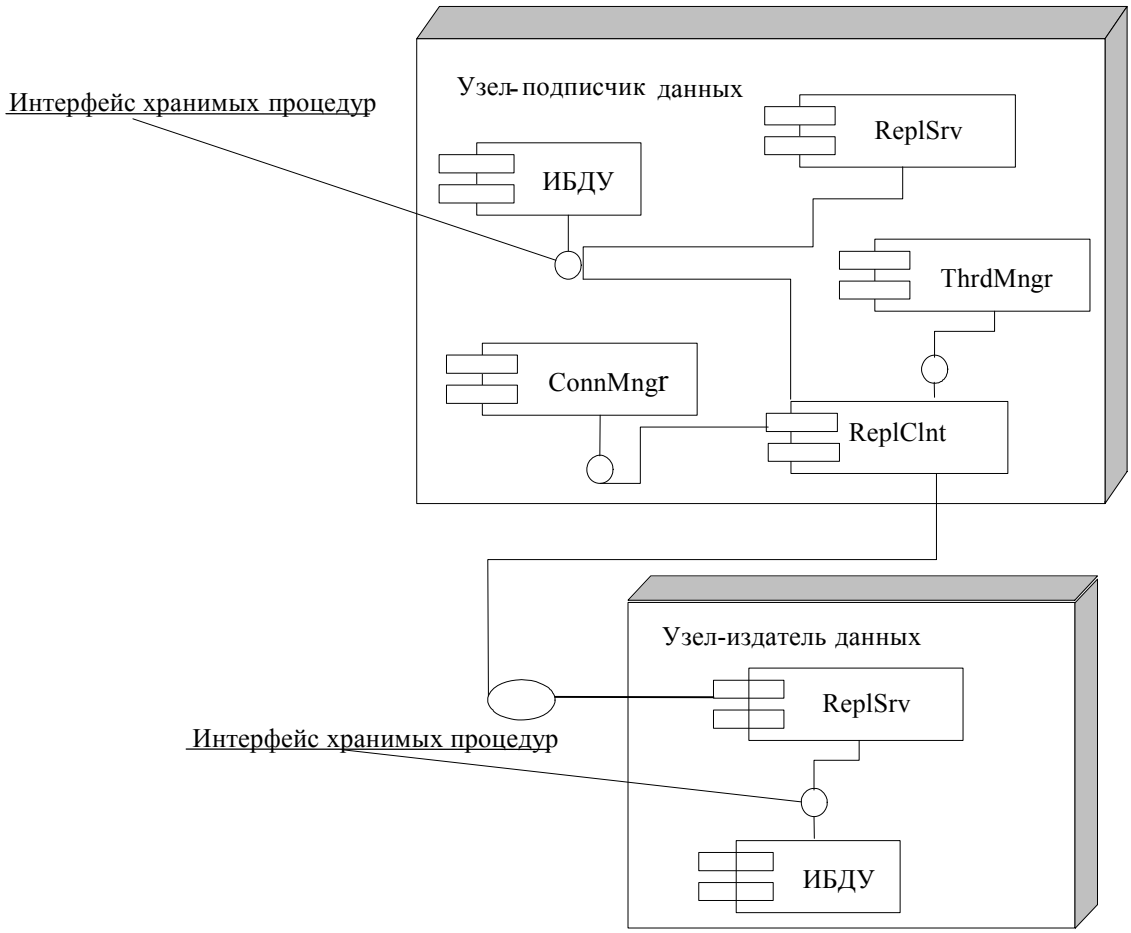

Рис. 3. Диаграмма размещения компонентов ПСД

Пересылаемые данные представлены в виде XML - структуры [11], что позволяет унифицировать кодирование данных на серверной стороне и их разбор со стороны клиента. Данные получаются серверной частью системы из ИБДУ в виде XML-файла с использованием выражения *FOR XML* языка Transact-SQL [2]. Клиентская часть использует объектную модель документов MS DOM для загрузки XML-файла, а для разбора и проверки правильности составленного XML-документа применяются функции MS XML Parser [10-11]. Пересланные данные репликации затем преобразуются с помощью имеющихся правил в строки вызова интерфейса *хранимых процедур* (*stored procedures*), обеспечивающих запись данных в ИБДУ. Эту задачу выполняет компонент *ReplClnt* (см. рис. 3).

Пересылаемые репликации содержат наименования параметров, их значения и момент времени получения данных на исходном объекте. Это ставит проблему, связанную с тем, что на различных узлах могут расходиться показания системных часов на терминалах. Если проводить синхронизацию по реальному значению времени (т.е. по показаниям каких-либо контрольных системных часов), то возникает проблема с согласованием хода часов на различных узлах МРИУС. Для ее устранения было принято решение сопоставить каждому сеансу съема параметров, с устройств узла, уникальный идентификатор (представленный полем в БД типа UNIQUEIDENTIFIER [2-3]), характеризующий данный конкретный момент времени, т.е. построить так называемую хронологическую модель репликации данных в БД, где единица времени - хронос - идентифицируется уникальным номером (кодом).

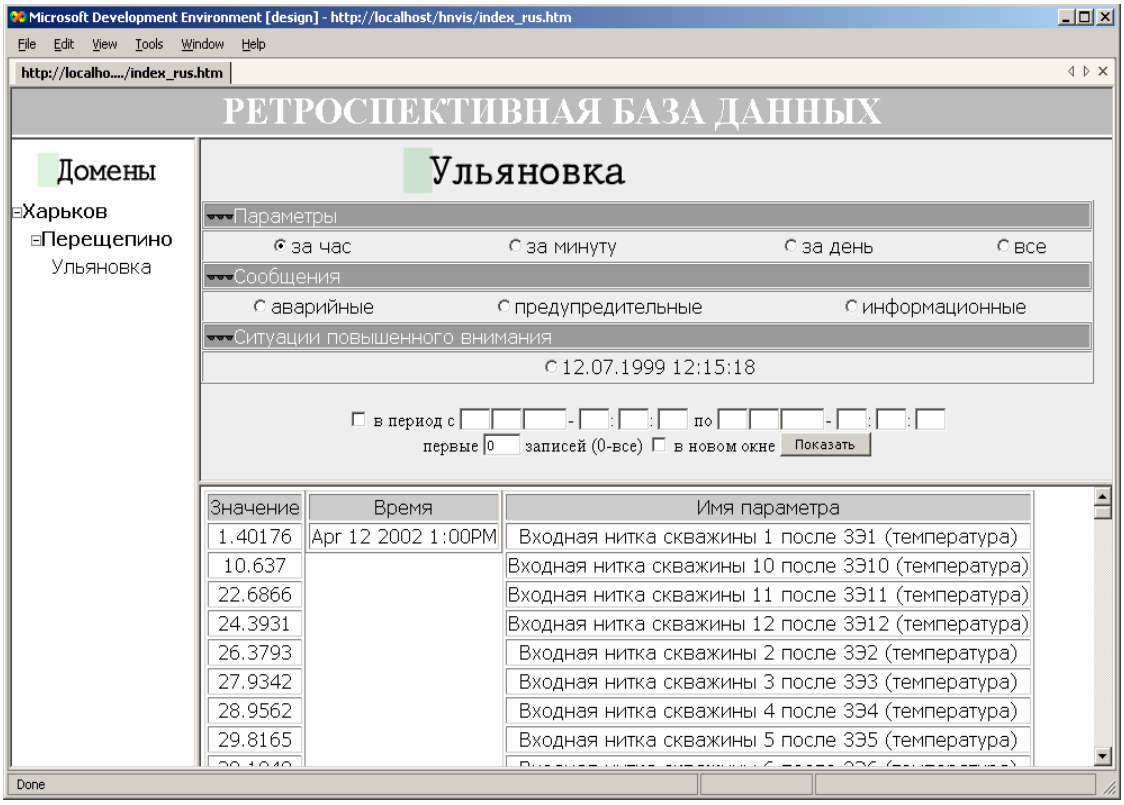

Рис. 4. Пример экранного интерфейса при работе с ПСД

Каждому моменту времени поставлены в соответствие показания системных часов узла - издателя (синхронизирующихся, в свою очередь, с часами сервера службами TCP/IP), на котором были получены эти параметры. Таким образом, на узле, которому данные будут пересланы (т.е. узлу-подписчику), будет известно, какие значения параметров соответствовали тем или иным показаниям системных часов на узле-издателе.

Взаимодействие клиентской и серверной частей ПСД могло бы быть организовано средствами технологий MS DCOM, CORBA или какой-либо другой объектно-ориентированной модели распределенного компонентного взаимодействия [3]. Однако из-за низкой пропускной способности реальных каналов связи в МРИУС, прежде всего телефонных, т.е. при скорости модемного соединения ниже 2400 bps архитектура COM/DCOM, использующая RPC (*Remote Procedur Call*) [3], работает нестабильно или не работает вообще.

Использование же предложенной Web-базированной архитектуры позволяет избежать проблем с плохой связью и возможностью помех и обрывов на линии, которые являются критическими при использовании RPC. Кроме того, применение протокола HTTP обеспечивает передачу данных в системе без потерь.

Заключение

В качестве возможных дальнейших программных решений для поставленной задачи синхронизации в МРИУС следует также рассмотреть новые технологии: *Web-Service Behavior* и *Simple Object Access Protocol (SOAP)* [10] для обеспечения работы сервиса синхронизации данных.

Технология SOAP позволяет сочетать преимущества удаленного вызова процедур RPC с возможностями работы через сетевое соединение с использованием протокола HTTP. Упрощенно суть SOAP заключается в том, что он действует так же,

как RPC, только вместо бинарной информации через соединение передается информация в формате XML с использованием HTTP. Это обеспечивает работу даже на нестабильных каналах связи, однако использование платформы MS .NET [10], на которой функционирует SOAP, все же требует больших вычислительных ресурсов на клиентской части такой системы.

Предложенную выше архитектуру уже можно рассматривать как упрощенную версию технологии SOAP, разработанную для конкретного класса задач синхронизации данных в Web-SCADA системах и успешно прошедшую тестирование в реальных условиях.

Литература

1. *Raynald M., Helary J.-M.* Synchronization and control of distributed systems and programs. - Kluwer Academic, 1996. – 156 p.

2. *Конолли Т., Бег К., Страчан А.* Базы данных: проектирование, реализация и сопровождение. Теория и практика. 2-е изд.: Пер. с англ. - М.- : Изд. «Вильямс», 2001.- 1120 с.

3. *Когаловский М.Р.* Энциклопедия технологий баз данных. – М.: Финансы и статистика, 2002. – 800 с.

4. *Калядин А.Ю.* SCADA-системы для энергетиков. ЗАО "РТСофт". –М., 2001. – URL: http://www.scada.ru/publication/.

5. SCADA системы в АСУ ТП. - «Объединение ЮГ». –К., URL:http://www.scada.com.ua.

6. Кукленко Д.В., Ткачук Н.В. Ретроспективная база данных в АСУ ТП: концепция и опыт реализации // Системний аналіз, управління і інформаційні технології: Вісник Харк. держ. політехн. ун-ту: Збірка наукових праць. – 2001.- Вип. 8.- Х., ХДПУ. - С. 62-68.

7. Potemkin V.V. Web- technologies at the Service of Process Control Systems // Вісник Національного технічного університету "ХПІ": Тематичний збірник наукових праць "Системний аналіз, управління та інформаційні технології". – Х., НТУ "ХПІ". - 2001. - № 22. - С. 23-27 (in English).

8. *Ткачук Н.В., Овасапов С.В. и др.* Разработка архитектуры региональной Web – базированной АСУ ТП для объектов газопромыслового управления «Харьковгаздобыча» // Вісник Національного технічного університету "ХПІ": Тематичний збірник наукових праць "Системний аналіз, управління та інформаційні технології". – Х., НТУ "ХПІ". - 2002. - № 9 (Т. 6). - С.51-60.

9. *Mykola V. Tkachuk, Heinrich C. Mayr et al.* Web-based Process Control Systems: Architectural patterns, Data Models, and Services // Lecture Notes in Computer Science (LNCS 2510), GI Edition, Berlin 2002. P. 721-729.

10. Microsoft Corporation Developer Network. – URL:http://msdn.microsoft.com

11. XML, RDF, DOM Specifications. – URL:http://www.w3.org.

Поступила в редакцию 14.04.03

Рецензенты: канд. техн. наук, доцент Минухин С.В., Харьковский национальный экономический университет, г. Харьков; д-р техн. наук, профессор, Годлевский М.Д., Национальный технический университет "ХПИ", г. Харьков.