УДК 004.414.22:006.065

ПРИМЕНЕНИЕ SOAP-СЕРВИСОВ ДЛЯ ОБЕСПЕЧЕНИЯ ВЗАИМОДЕЙСТВИЯ ВНУТРИ РАСПРЕДЕЛЕННОЙ ИНФОРМАЦИОННОЙ СИСТЕМЫ И УДАЛЕННОГО УПРАВЛЕНИЯ РЕСУРСАМИ

Д.Н. Дутиков

SOAP-SERVICES USING FOR DISTRIBUTED INFORMATION SYSTEM COLLABORATING AND RESOURCES REMOTE MANAGEMENT

D.N. Dutikov

В работе на примерах рассматриваются способы применения SOAP-сервисов в практике удаленного управления ресурсами интрасети, даются общие рекомендации по созданию элементов информационной системы, отвечающих за взаимодействие с устройствами. Приводится пример выбора архитектуры построения информационной системы и сравнительный анализ средств программирования применительно к задаче взаимодействия с устройствами.

Ключевые слова: протокол SOAP, удаленное взаимодействие, информационные системы, автоматизация сбора данных, распределенные системы.

In samples described SOAP-services using methods in an intranet remote resource management practice. Proposed common solutions for device interaction modules of information systems. Sample of common architecture selection to solving problem of device interaction in the web produced.

Keywords: SOAP protocol, collaboration and remote interaction, information systems, automation of data collection, distributed systems.

Введение

В ходе создания системы мониторинга данных геолого-минералогических исследований на Южном Урале (далее - информационная система) был решен ряд сложных задач. Одной из них была задача организации удаленного доступа к приборной исследовательской базе и автоматизация сбора данных.

С ростом числа элементов системы повышаются затраты на её поддержку. В случае необходимости поддерживать связь сразу с несколькими приборами, эти затраты могут превысить возможности информационного отдела в исследовательском институте. Чтобы не прийти к такой ситуации необходимо заранее проводить серьезное планирование системы в целом [1]. Центральное место в этом планировании занимает выбор платформы, языков программирования и технологий взаимодействия.

На сегодняшний день можно найти множество информации по планированию информационных систем [2-4] и даже по выбору языков программирования [5]. Существует также опыт интеграции в общую систему промышленного оборудования и установок физического профиля [4, 6]. Однако в большинстве своем авторы затрагивают темы взаимодействия с устройством больше, чем протоколов взаимодействия внутри системы. Между тем, отмечается, что современная информационная система должна быть построена с использованием открытых стандартов и протоколов [2, 3, 7].

В данной работе предлагается один из возможных вариантов организации удаленного взаимодействия в рамках научно исследовательского института с использование протокола SOAP.

1. Intranet/internet-технологии при построении информационной системы

Информационная система построена на базе web-технологий, обеспечивающих хорошее масштабирование, понятный и прозрачный процесс администрирования. Web-технологии также позволяют достаточно просто организовывать удаленный доступ к вычислительным ресурсам, вместе с

Дутиков Дмитрий Николаевич - ведущий программист отдела геоинформационных технологий Института минералогии УрО РАН; odoom@mineralogy.ru

Dutikov Dmitry Nikolaevich - leading software developer of Geoinformation Technologies Department of Institute of Mineralogy of The Ural Branch of RAS; odoom@mineralogy.ru

тем, они не требуют специфичного клиентского программного обеспечения, таким образом, достигается кроссплатформенность всей системы в целом. Доступ к такой системе возможен как по локальной вычислительной сети, так и через каналы Internet, что на сегодняшний день немаловажно.

Web-технологии появились достаточно давно, и на сегодняшний день существуют способы распределения процессов обработки пользовательских запросов на несколько серверов (web-фермы). В перспективе это обеспечивает организацию стабильной работы информационной системы при большом количестве одновременных запросов.

Несмотря на то, что построение информационной системы с использованием web-технологий во многих отношениях предстает перед нами в выгодном свете, вполне очевидны два главных недостатка такого подхода. Первым отметим вопрос безопасности. Действительно, вопрос безопасности волнует многих пользователей информационной системы, поскольку предполагается хранение в ней данных научных исследований, утрата которых или искажение недопустимы, так же, как и нежелательно преждевременное оглашение данных (по вполне понятным причинам) [9]. Вопросы безопасности достаточно сложны и требуют отдельного рассмотрения. Впрочем, заметим, что при разработке информационной системы мы учли основные способы «взлома» web-систем (такие как sql-инъекции, кроссдоменный скриптинг, подмена аутентификационного cookie).

Второй проблемой мы отметим возможности языков программирования и библиотек классов, используемых при создании web-систем. Информационная система спланирована не только как хранилище, банк данных, предоставляющий функции сбора данных, поиска и представления. Как было отмечено в начале, система должна также и сама, в автоматическом режиме, взаимодействовать с приборами. В необходимости такого взаимодействия и кроется проблема. В каждом случае, к каждому прибору необходим индивидуальный подход. Способы взаимодействия с прибором в большей степени относятся к разделу системного программирования, а не web-программирования. Гетерогенность системы, таким образом, повышается, увеличиваются затраты на её поддержку.

Приведем наше решение проблемы, сводящее к минимуму количество технологий, языков программирования и способов взаимодействий внутри информационной системы.

2. Организация удаленного взаимодействия

Наша информационная система, как и любая другая, располагается и работает на какой-то определенной серверной машине. Сложно себе представить, что вся приборная база будет каким-то образом подключена к этому компьютеру. Хотя существуют технологии удаленного доступа к аппаратным портам компьютера (например COM port redirector), использование их возможно не во всех случаях. Например, программное обеспечение к какому-либо устройству может быть жестко привязано к конкретной машине и будет работать только на ней. Да и администрирование сервера со множеством различного программного обеспечения - задача сложная, могут так же возникать проблемы с безопасностью.

Наиболее правильным нам видится использование рабочих станций в том режиме, в котором они уже работают, и с такой конфигурацией, какая представляется производителям программного обеспечения наиболее правильной. Таким образом, возможно обеспечить бесперебойную работу программного обеспечения на местах.

Удаленное взаимодействие можно обеспечить при помощи внедрения в рабочую станцию агента информационной системы, отслеживающего состояние прибора и отсылающего сведения на центральный сервер. Такую схему можно реализовать несколькими более или менее стандартными средствами:

- при помощи технологий COM или CORBA, при этом на рабочей станции запускается объект, который в теории обладает доступом ко всем ресурсам компьютера;
- при помощи технологий .NET-remoting и WCF;
- при помощи клиентского сервиса, исполняемого на стороне рабочей станции и по определенному протоколу отсылающего данные на центральный сервер.

При выборе способа взаимодействия всегда необходимо учитывать, что архитектура системы может измениться, может возникнуть потребность в ее перестройке или объединении с другой системой. Интероперабельность должна достигаться в том числе и за счет использования стандартных и открытых протоколов [7]. Для обеспечения максимально легкого взаимодействия в разнородной среде был выбран протокол SOAP.

Протокол SOAP легко реализуем в различных системах, его поддерживают многие платформы программирования. Мы не будем приводить всех тонкостей протокола и детального его описания, они доступны во многих открытых источниках. Скажем лишь, что это протокол обмена структурированными сообщениями. Основная задача протокола - поддержка передачи данных со строгим описанием типов и сложной структурой, что позволяет на его основе организовывать удаленный вызов процедур. Протокол этот работает поверх протокола HTTP, поэтому возможно объединение в одну систему машин, находящихся в разных сегментах сети и за системой NAT. Использование HTTP в качестве базового протокола облегчает задачи обеспечения безопасности, поскольку нет необходимости держать на сервере открытыми какие-либо дополнительные порты.

Предлагается следующая общая технология организации взаимодействия с прибором.

1. На рабочих местах, под управлением которых находится прибор, устанавливается SOAP-сервер. Наиболее простой способ его организации - это HS и ASP.NET (в операционной системе Windows) или Apache и любая технология, поддерживающая SOAP (Perl, PHP, Phyton, Mono и т. п.).

2. SOAP-сервис, посредством COM, TCP/IP, CLI, API или любой другой доступной технологии, получает данные и отправляет их на сервер.

3. На серверной стороне информационной системы так же существует SOAP-сервис, задача которого - принимать данные.

3. Возможности языков web-программирования и потребности информационной системы

Для сведения к минимуму разнородности системы необходимо, чтобы клиентская часть системы и серверная были созданы при помощи одинаковых технологий и языков программирования. Поэтому очень важен первоначальный выбор языка. Подробно свой выбор мы обосновывали в работе [8], теперь же приведем основания, связанные с необходимостью взаимодействия с приборной базой.

Нужно сразу сделать оговорку. Современные языки программирования высокого уровня, как правило, обладают схожими возможностями. Из различий, в первую очередь, видны синтаксические, строгость типизации; встречаются различия в применяемой парадигме программирования (языки ООП, функциональные языки, объектные языки). Однако, в конечном итоге, всякие различия между языками никак не отражаются на результате работы программиста.

Со специализированными языками программирования и языками среднего уровня дело обстоит сложнее, их выбору посвящают отдельные работы (см. [5]).

Правильней было бы сравнивать библиотеки классов (или аналогичные библиотеки, предоставляющие программисту необходимый набор функций), но сейчас они настолько тесно «срослись» с поддерживаемыми языками программирования, что едва ли возможно их упоминание по отдельности. Поэтому когда теперь мы пишем о «языках программирования», следует подразумевать так же и основной набор функций, употребляемый вместе с этим языком.

Из наиболее широко применяемых сегодня языков мы выделим PHP, Java, C# (ASP.NET). Все три языка являются языками высокого уровня, поддерживающими парадигму объектно-ориентированного программирования.

Язык PHP - интерпретируемый язык, с обширной библиотекой функций, в которой содержатся все функции, необходимые для работы web-программиста, начиная от операций с текстом и регулярными выражениями до работы с базами данных. Так же существуют дополнительные объектные структуры (frameworks), реализующие каркас приложения, что сильно ускоряет разработку приложения.

Java и C# (си шарп) - это управляемые языки программирования. Оба поддерживаются обширнейшими библиотеками классов, в которых можно найти классы, что называется, «на все случаи жизни». Оба языка используются как для создания web-приложений, так и для создания настольных приложений. Существуют объектные структуры для этих языков и готовые решения.

Из всего многообразия возможностей выделим необходимые нам для взаимодействия с аппаратной базой. Обычно приборы поставляются с некоторым программным обеспечением. Подавляющее число программного обеспечения к приборам работает только в среде операционной системы Windows. Хорошо это или плохо - вопрос отдельный. Нас интересует только техническая сторона вопроса. А она такова, что в некоторых случаях у программного обеспечения (драйверы, настольные приложения для оператора) существует программный интерфейс (API). Программный интерфейс может заключаться в простых динамически загружаемых библиотеках (dll) или библиотеках, поддерживающих общую модель объектов (COM). Не исключаем также возможность существования статических библиотек под системы Unix/Linux. Менее удобным может оказаться наличие интерфейса командной строки (CLI - command line interface) - ряда утилит, работающих в рамках стандартных выводных и входных потоков (stdin, stdout, stderr потоки). Еще одной возможностью взаимодействия является взаимодействие по протоколу tcp/ip, когда само устройство или программное обеспечение к нему поддерживает соединения на определенном порту и принимает по нему команды. Еще одним очевидным способом взаимодействия является прямое взаимодействие с устройством через аппаратные порты.

Итак, нами было выделено несколько возможных способов взаимодействия с аппаратным обеспечением:

- прямое взаимодействие с устройством;
- взаимодействие через API в виде dll;
- через COM-объекты;
- по протоколу TCP/IP;
- посредством утилит командной строки (см. также [6]).

Теперь, зная необходимую нам функциональность, обратим внимание на перечисленные языки программирования.

При рассмотрении таблицы можно заключить, что наиболее популярный язык программирования для создания web-приложений - PHP наименее пригоден для реализации поставленной задачи. Несмотря на огромную функциональность, связанную с web-разработкой, этот язык не обладает набором функций системного программирования. Ограниченная поддержка COM еще сильней усугубляет таковое положение.

Поддержка требуемой функциональности в языках web-программирования

Функция	Поддержка в PHP	Поддержка в C# (ASP.NET)	Поддержка в Java
Доступ к COM, LPT, USB портам	Нет	Параллельные порты – встроенная поддержка, последовательные и USB сторонними компонентами	Поддерживается
Раннее связывание DLL	Нет	Поддерживаются	Поддерживается
Позднее связывание DLL	Нет	Поддерживается через раннее связывание с WIN API (LoadLibrary, GetProcAddress, FreeLibrary)	Поддерживается
Взаимодействие с COM	Поддерживается ограничено. Объекты создаются только в рамках одного скрипта	Поддерживается	Поддерживается через JNI (вызов стандартных функций из библиотек COM) или JACOB (Java-COM bridge, использует JNI)
Поддержка tcp/ip sockets	Поддерживается	Поддерживается	Поддерживается
Управление дочерним процессом	Частичное	Поддерживается	Поддерживается
Управление стандартными потоками дочернего процесса	Нет	Поддерживается	Поддерживается
Кроссплатформенность	Полная	Частично, через Mono	Полная

Между конкурирующими платформами Java и .NET мы находим мало различий, как минимум вся необходимая функциональность в них так или иначе присутствует. Основным языком программирования был выбран C# и платформа ASP.NET. Выбор такой обусловлен необходимостью обеспечения преемственности с уже существующей, но устаревшей информационной системой в Институте минералогии УрО РАН (г. Миасс). Старая система была написана на ASP (Active Server Pages), в результате сложился и устоялся определенный состав серверного программного обеспечения, был сделан выбор определенной СУБД, произведено обучение и администрирующего персонала. Эти и некоторые другие причины [8] стали основанием сделанного выбора.

4. Ситуативный выбор средств взаимодействия с прибором

Несмотря на описанную нами общую структуру взаимодействия с прибором и последующей передачи данных в хранилище, конечный выбор структуры системы должен быть подвержен здравому смыслу. Основным направлением оптимизации системы должно быть сокращение числа промежуточных операций там, где это возможно.

Рассмотрим конкретный пример. Существует датчик температуры DS9097U, подключаемый к последовательному порту компьютера. Задача: обеспечить автоматический сбор данных с этого датчика через каждые 5 минут. Данные должны быть записаны в отдельную таблицу общей базы данных.

По описанной выше логике, на рабочей станции, к которой подключен датчик, должна быть установлена клиентская система. В нашем случае - это IIS и web-сервис, опрашивающий последовательный порт для получения данных с датчика. Сервер через каждые пять минут должен вызывать метод сервиса для получения этих данных и записи в базу.

На практике оказывается, что датчик подключен к компьютеру, на котором располагается собственно СУБД, что дает возможность производить запись в базу, минуя всю схему с сервисами SOAP. Кроме того, для датчика написана консольная утилита, снимающая показания датчика и выводящая сообщение на стандартный выводной поток в удобном для человека виде. В таком случае для получения данных проще запускать эту утилиту, а ее вывод анализировать при помощи машины состояний или регулярных выражений.

Для решения поставленной задачи был выбран упрощенный путь. Создана консольная утилита на языке C#, которая запускает программу для получения показаний датчика и записывает полученные данные в базу. Запуск созданной консольной утилиты был назначен в системном планировщике заданий через каждые пять минут.

Таким образом, на плечи центральной информационной системы ложится лишь отображение данных, уже находящихся в таблице, в виде графиков. С результатами работы можно ознакомиться по адресу http://monitoring.mineralogy.ru/temperature.

Рассмотрим еще один пример. В локальной сети существует фильтрующий прокси-сервер под управлением операционной системы FreeBSD; задача сервиса - ограничивать входящий internet-трафик по набору правил. Необходимо предоставить администратору удобный интерфейс для поиска некорректно работающих правил с функциями удаления и редактирования найденных правил.

Хотя этот пример не относится напрямую к системе взаимодействия с аппаратными средствами, подход к решению проблемы остается в точности таким же. На стороне фильтрующего прокси-сервера поднимается httpd-сервис Apache и система Mono. Под управлением Apache создается SOAP-сервис, реализующий функции поиска и редактирования правил, переконфигурации прокси-сервера. На стороне информационной системы создается модуль, предоставляющий интерфейс к SOAP-сервису и дополнительные функции, обеспечивающие удобство работы и контроль за правами доступа. Методы SOAP-сервиса вызываются с серверной стороны информационной системы.

Из примера видно, что несмотря на различия в операционных системах, информационная система по-прежнему подвержена общим стандартам взаимодействия и написана на одном языке программирования (C#). Однако заметим, что функции SOAP-сервиса можно вызывать напрямую со стороны этого сервиса и там же предоставлять интерфейс. Описанная схема создана для обеспечения безопасности, при таком подходе возможно ограничить доступ к самому сервису по единственному адресу.

В следующем примере рассмотрим ситуацию, когда полностью автоматическая работа системы невозможна или не нужна. Существует устройство для проведения рентгеноструктурного анализа образцов вещества. К устройству поставляется программное обеспечение. По описанному нами алгоритму возможно, конечно, создать автоматическую систему сбора, но для этого есть существенное препятствие. Анализ каждый раз проводится единично. Оператор должен загрузить в прибор заранее подготовленный препарат, нажать кнопку, провести оценку качества работы прибора. Только после этого возможно внесение данных в базу. В результате этого, вся описанная схема взаимодействия сразу теряет смысл. Впрочем, в некоторой степени автоматизировать работу оператора все же можно. Данные, выдаваемые программным обеспечением, записываются в файл. Вместо того чтобы вручную заносить все полученные данные, оператор может загрузить файл, а информационная система сама извлекает из него данные и записывает в базу.

Заключение

Итак, описанные нами возможности интеграции информационной системы путем унификации протоколов взаимодействия, использования открытых протоколов взаимодействия является перспективным направлением разработки в области обеспечения удаленного доступа к аппаратному обеспечению исследований и обеспечению автоматического и автоматизированного сбора информации.

Приведенные примеры показывают основные схемы построения системы автоматического сбора информации и управления ресурсами. В каждом отдельном случае, с каждым отдельным прибором проблема должна решаться индивидуально, однако следует помнить, что в конечном итоге доступ к прибору должен осуществляться по открытому протоколу.

В перспективе планируется использовать протокол SOAP и стандарт WSDL для организации удаленного взаимодействия с приборами, автоматической настройки передачи потокового видео получаемого с прибора.

Важно отметить, что сам по себе протокол SOAP является надстройкой над XML и HTTP, скорость обработки запросов по этому протоколу ниже, чем скорость обработки «чистых» HTTP запросов. Поэтому в системах, где время отклика является критической характеристикой, от идеи использовать данный протокол следует отказаться.

Литература

1. Построение интегрированного информационного пространства предприятия / Е.Н. Ишметьев, Ю.Н. Волщуков, А.В. Романенко и др. // Управление информационной инфраструктурой современной организации на основе технологии открытых систем: сб. тр. IV междунар. науч.-практ. семинара. — Магнитогорск: МаГУ, 2006. — С. 54-59.

2. Вендров, А.М. CASE-технологиии: современные методы и средства проектирования информационных систем. /А.М. Вендров — М.: Финансы и статистика, 1998. - 176 с.

3. Клещев, Н.Т. Практическое руководство по созданию и проектированию информационных систем /Н.Т. Клещев, А.А. Романов. - М.: Изд-во ООО «Научтехиздат», 2001. - 389 с.

4. Калинин, Э.О. Практические рекомендации по разработке и внедрению технологии распределенных систем на промышленном предприятии / Э.О. Калинин // Система обработки информации и управления: архитектура и программное обеспечение: сб. науч. тр. - Челябинск: Изд-во ЮУрГУ, 1998. - С. 14-19.

5. Сидельников, В.И. Методика выбора языка программирования для реализации программного продукта / В. И. Сидельников // Автоматизация и современные технологии. — 2007. — № 7. — С. 27—32.

6. Куликов, Г.Э. Автоматизированная система сбора и обработки данных для установок фи-

зического профиля / Г. Э. Куликов // Информационные технологии. — 2008. -№4. -С. 50-57.

7. Олейников, А.Я. Состояние и перспективы развития технологии открытых систем / А.Я. Олейников, Т.Д. Штробокова // Управление информационной инфраструктурой современной организации на основе технологии открытых систем: сб. тр. IVмеждунар. науч.-практ. семинара. -Магнитогорск: МаГУ, 2006. - С. 41-46.

8. Дутиков, Д.Н. Использование новейших технологий для хранения обработки и представления данных геолого-минералогических исследований /Д.Н. Дутиков // Наука и технологии. Итоги диссертационных исследований. - М.: РАН, 2009. -Т. 2.- С. 17-29. - (Избранные труды Российской школы).

9. Гусев, М.О. Вопросы защиты информации в профиле академического института / М. О. Гусев, А.Я. Олейников // Управление информационной инфраструктурой современной организации на основе технологии открытых систем: сб. тр. IV междунар. науч.-практ. семинара. — Магнитогорск: МаГУ, 2006. - С. 74-81.

Поступила в редакцию 28 января 2010 г.