УДК 004.057.3

Ф. И. Кушназаров, В. В. Яковлев, О. А. Турдиев

СРАВНЕНИЕ ПРОИЗВОДИТЕЛЬНОСТИ ПРОТОКОЛОВ ДОСТУПА К ОБЛАЧНЫМ РЕСУРСАМ

Дата поступления: 23.11.2015
Решение о публикации: 28.12.2015

Цель: Дать сравнительную оценку производительности протоколов доступа к облачным ресурсам REST и SOAP, выявить эффективность протокола относительно выборки, запросов и других параметров. **Методы:** Протоколы сравнивали с помощью математических методов, результаты проверяли в Java-классах. Использовали простые запросы в HTTP методами GET и POST. **Результаты:** Подробно описаны протоколы доступа REST и SOAP к облачным ресурсам – принцип работы, формат сообщения, используемые приложения. Приведены результаты сравнения этих протоколов, выявлена производительность каждого протокола в одиночных и множественных запросах, по результатам производительности запросов определена эффективность протоколов доступа к облачным ресурсам. Также определены факторы, влияющие на производительность протоколов доступа к облачным ресурсам (помехозащищенность соединения, скорость передачи данных). **Практическая значимость:** Результаты анализа позволяют определить производительность протоколов, а также их эффективность в зависимости от области применения.

Протоколы, производительность, эффективность, компьютерные сети, облачные технологии, доступ, REST, SOAP.

Farrukh I. Kushnazarov, postgraduate student, k.farruh@bk.ru; **Valentin V. Yakovlev**, D. Eng., professor, jakovlev@pgups.ru; ***Odilzhan A. Turdiyev**, master, odiljan.turdiev@mail.ru (Petersburg State Transport University) COMPARISON OF PERFORMANCE OF CLOUD RESOURCES' ACCESS PROTOCOLS

Objective: To provide a comparative evaluation of performance of REST and SOAP access protocols for cloud resources, to determine protocol efficiency concerning access, requests and other parameters. **Methods:** Protocols were compared with the use of mathematical methods, results were checked in Java classes. Simple requests in HTTP by GET and POST methods were used. **Results:** REST and SOAP protocols for cloud resources access were described in detail – operating principle, message format, applications used. Results of comparison of these protocols provided, productivity of each protocol in single and multiple requests established, by results of request productivity cloud resource access protocols' efficiency was established. Factors that influence productivity of cloud access protocols were also established (communication interference protection, data flow speed). **Practical importance:** Analysis results allow to evaluate protocols' productivity, as well as their efficiency depending from application field.

Protocols, performance, efficiency, computer networks, cloud computing access, REST, SOAP.

Архитектура REST

Одной из проблем облачных вычислений является выбор протоколов доступа: FTP (обычно используется для доступа к облачным хранилищам), WebDAV (расширение протокола HTTP), REST и SOAP. У каждого из них есть своя область применения, досто-

инства и недостатки. В данной статье с теоретической точки зрения рассмотрены технологии REST и SOAP.

REST (Representational State Transfer – передача репрезентативного состояния) – это не стандарт и не спецификация, а архитектурный стиль, выстроенный на существующих хорошо известных стандартах, таких как HTTP, URI и XML. В REST-сервисах акцент сделан на доступ к ресурсам, а не на исполнение удаленных сервисов. В общем случае REST является очень простым интерфейсом управления информацией без использования каких-то дополнительных внутренних прослоек.

Термин REST ввел в 2000 г. Рой Филдинг, один из разработчиков протокола HTTP [5, 6], в качестве названия группы принципов построения веб-приложений. Филдинг описал концепцию построения распределённого приложения, при которой каждый запрос (REST-запрос) клиента к серверу содержит в себе исчерпывающую информацию о желаемом ответе сервера, и сервер не обязан сохранять информацию о состоянии клиента («клиентской сессии») [3].

Управление информацией сервиса целиком и полностью основывается на протоколе передачи данных. Наиболее распространен протокол HTTP, но в целом REST охватывает более широкую область, нежели HTTP: его можно применять и в других сетях с другими протоколами. REST описывает принципы взаимодействия клиента и сервера, основанные на понятиях «ресурса» и «глагола». В случае HTTP ресурс определяется своим URI, а глагол – это HTTP-метод.

Как уже сказано, REST определяет архитектуру «клиент – сервер», в которой клиенты получают доступ к ресурсам сервера посредством представления, экспортируемого сервером. Клиенты обращаются через единый интерфейс не прямо к ресурсам, а к их представлению. Как и многие другие типы архитектуры «клиент – сервер», REST реализуется как многоуровневая архитектура, что позволяет использовать различные возможности, предоставляемые нижними уровнями (например, выравнивание нагрузки HTTP и т. п.).

REST-сервисы достаточно просто реализовывать, поскольку они базируются на хорошо известных протоколах и не требуют от разработчика изучения разных WS-спецификаций. Самый простой REST-сервис можно создать за несколько минут: статичный XML-файл, возвращаемый веб-сервисом, это технически и есть REST-сервис, так как XML-данные запрашивались через HTTP. По виду пришедшего запроса сразу можно определить, что он делает, не разбираясь в форматах (в отличие от SOAP, XML-RPC). Данные передаются без применения дополнительных слоев, поэтому REST считается менее ресурсоемким: нет необходимости разбирать запрос, чтобы понять, что именно он должен сделать, не надо переводить данные из одного формата в другой.

Типы запросов REST

Так как разработка сервисов обычно ведется с использованием HTTP, следует использовать методы этого протокола для определения действия над ресурсом. Общеприняты следующие стандарты:

- **GET/URL/**(Index) – получение списка всех объектов. Как правило, это упрощенный список, т. е. содержащий только поля идентификатора и названия объекта, без остальных данных;
- **GET/URL/{id}** (View) – получение полной информации об объекте;
- **POST/URL/**(Create) – создание нового объекта. Данные передаются в теле запроса без применения кодирования;
- **PUT/URL/{id}** (Edit) – изменение данных с идентификатором {id}, возможно, замена данных;
- **DELETE/URL/{id}** (Delete) – удаление данных с идентификатором {id}.

Протокол SOAP

SOAP (Simple Object Access Protocol – простой протокол доступа к объектам), расшире-

ние протокола XML-RPC [2], который, в свою очередь, является одним из стандартов вызова удаленных процедур, использующий XML для кодирования сообщения и HTTP в качестве транспортного механизма.

XML-RPC. Краткие сведения

XML-RPC (Extensible Markup Language Remote Procedure Call – XML-вызов удалённых процедур) – протокол вызова удаленных процедур, который, как и любой другой интерфейс RPC, определяет набор стандартных типов данных и команд, используемых для доступа к функциональности программы, находящейся удаленно. Формат XML позволяет унифицировать передачу данных между приложениями, а использование в качестве транспорта HTTP снимает ограничения, налагаемые как на конфигурацию сети, так и на маршрут следования пакетов. Вызовы XML-RPC представляют собой простой тип данных text/xml и свободно проходят сквозь шлюзы везде, где допускается ретрансляция http-трафика [1].

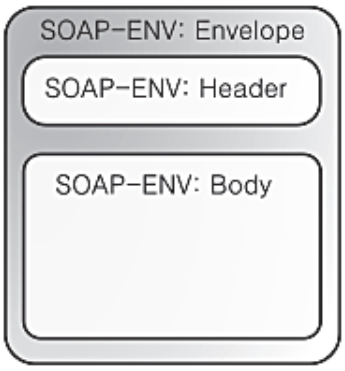
Как следует из названия данного протокола, в его основе лежит главная идея RPC – сделать вызов удаленной процедуры выглядящим по возможности так же, как вызов локальной процедуры. По-другому это можно сформулировать следующим образом: вызывающей процедуре не требуется знать, что вызываемая процедура находится на другой машине, и наоборот. Это достигается с помощью процедур-заглушек (от англ. stub – заглушка). Клиентское приложение, запрашивающее вызов удаленной процедуры, на самом деле вызывает процедуру-заглушку, список параметров которой в точности соответствует вызываемой процедуре. Затем заглушка преобразует параметры в соответствующий пакет, предназначенный для передачи по сети серверу. На стороне сервера имеется такая же заглушка, которая выполняет обратную процедуру: сетевой пакет преобразуется в список параметров, вызывается и выполняется требуемая локальная процедура, результат ее выполнения передается в процедуру-заглушку, которая формирует пакет для передачи в сеть. В конце концов, на стороне клиента этот пакет преобразуется в итоговый результат.

В XML-RPC для передачи используются девять типов данных: семь простых и два сложных. К простым относятся целые и вещественные числа, дата и время, логический тип и строки с кодировкой ASCII или base64. Сложные типы представлены структурами и массивами, причем массивы могут быть многомерными. Также поддерживается обработка ошибок: так, если при обработке запроса произошла ошибка, то в ответе будет элемент <fault>, в который будет вложена структура, содержащая числовой код ошибки и ее содержание.

Данный протокол является достаточно простым и идеально подходит для построения сервисов, поддерживающих передачу данных и не нуждающихся в сложных командах. Для систем же, работающих со сложной логикой и для передачи больших комплексных структур данных, XML-RPC является не лучшим вариантом. Именно поэтому компания Microsoft решила расширить функциональность этого протокола, что привело к появлению нового стандарта – SOAP.

Формат сообщений

SOAP-конверт содержит либо запрос на осуществление некоторого действия, либо



Структура SOAP-сообщения

ответ – результат выполнения этого действия. Конверт и его содержимое закодировано языком XML. Структура SOAP-сообщения приведена на рисунке, где конверт (Envelope), заголовок (Header) и тело (Body) являются основными частями SOAP-сообщения.

Описание работы приложения

В ходе работы были написаны два клиент-серверных приложения «Таблица факториалов» по технологиям REST и SOAP. Данные приложения выполняют аналогичные функции (добавление, просмотр, поиск, удаление), но реализованы они по-разному. Клиент обращается к серверу, на котором происходят все необходимые вычисления, затем серверные части приложений разворачиваются в облаке Microsoft.

Для написания приложений было использовано такое ПО, как JDK 1.7, Windows AzureSDK, IDEEclipse 4.4.1, плагин Windows AzureToolkitforEclipse, сервер ApacheTomcat 7.0.47, IDENetBeans 7.3.1 (для клиентской части), а также APIJAX-RS и JAX-WS. Также необходим действующий аккаунт Windows Azure.

REST-приложение

Приложение по технологии REST написано с помощью фреймворка Jersey. В дескрипторе развертывания web.xml в качестве класса сервлета указан стандартный класс из библиотеки Jersey, также указан пакет, в котором Jersey будет искать классы веб-сервиса, и URL-путь к веб-сервису.

Так как REST основан на HTTP, при передаче ответа от сервера к клиенту используются стандартные коды HTTP.

SOAP-приложение

В отличие от REST, SOAP не может использовать при ответе коды ошибок HTTP. Вместо этого в SOAP-сообщении существует элемент Fault, предоставляющий информацию о проблеме [7]. Помимо стандартных ошибок, предусмотренных спецификацией, можно создавать и возвращать собственные объекты Fault.

Серверная часть приложения состоит из трех классов и одного интерфейса:

• класса-сущности Fact, обычного класса с двумя полями: числом и его факториалом;

• класса-модели Factory, в котором описаны методы для работы с объектами типа Fact;

• интерфейса FactService, в котором объявлены методы взаимодействия между клиентом и сервером;

• класса FactImpl, реализующего методы, объявленные в FactService.

Сравнение производительности RESTи SOAP

Производительность сравнивали путем измерения времени аналогичных запросов SOAP и REST. Время запросов измеряли с помощью методов класса java.util.Date. Методы, для которых сравнивалась производительность, – HTTP-методы GET и POST для REST-приложения и методы GetAllFacts () и AddFacts (List<Fact>fact) для SOAP-приложения. Таким образом проверяли запросы для добавления и получения объектов.

Производительность рассчитывали следующим образом: неоднократно измеряли время одних и тех же запросов и вычисляли среднеарифметическое значение этих запросов по формуле

$$\overline{x} = \frac{x_1 + x_2 + \ldots x_N}{N},$$

где x_i – однократное время выполнения действий; N – объем выборки.

Затем высчитывали среднеквадратичное отклонение:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{n}(x_i - \overline{x})^2},$$

где $\overline{x}$ – среднеарифметическое значение и временную эффективность:

$$\delta = 100 - \frac{t_R + \sigma_R}{t_S + \sigma_S} \cdot 100,$$

где t_R и t_S – время выполнения REST- и SOAP-запросов, соответственно; σ_R и σ_S – среднеквадратичное отклонение REST- и SOAP-сервисов, соответственно.

Результаты измерений представлены в табл. 1–3.

Также было вычислено количество байт, передаваемых при ответе от сервера к клиенту. Вычисления производили в программе SOAPUI, их результаты представлены в табл. 4.

По итогам измерений можно сделать вывод, что REST-сервис имеет преимущество в производительности над аналогичным SOAP-сервисом. Это выражается как в меньшем времени запросов к серверу от клиента, так и в размере передаваемых сообщений. При этом следует отметить, что если бы REST-сервис передавал сообщения не в формате XML,

ТАБЛИЦА 1. Результаты одиночного запроса

Тип запроса	Добавление запроса, мс	Получение запроса, мс
REST	468 ± 38,4	680 ± 57,7
SOAP	481 ± 42,6	703 ± 68,1

ТАБЛИЦА 2. Эффективность REST по сравнению с SOAP

Операция	Производительность, %
Одиночное добавление	3,28
Одиночное получение	4,33
Множественное добавление	7,29
Множественное получение	9,17

ТАБЛИЦА 3. Результаты множественного запроса

Тип запроса	Количество запросов	Добавление запроса, мс	Получение запроса, мс
REST	50	1084 ± 97,1	1380 ± 111,8
	100	1294 ± 104,7	1881 ± 167,6
	250	1852 ± 126,6	2974 ± 255,8
	500	2301 ± 197,4	4598 ± 343,2
	1000	3045 ± 292,9	7337 ± 696,4
SOAP	50	1174 ± 106,8	1512 ± 132,7
	100	1378 ± 117,2	1974 ± 183,4
	250	1997 ± 145,1	3442 ± 290,7
	500	2568 ± 212,5	5174 ± 378,4
	1000	3289 ± 313,7	8217 ± 710,5

ТАБЛИЦА 4. Размер передаваемых сообщений

Тип запроса	Количество запроса	REST, байт	SOAP, байт	Разница, байт
Получение запроса	1	122	269	147
	50	5065	5310	245
	100	13971	14316	345
	250	84307	84952	645
	500	379994	381139	1145
	1000	139591	1398076	2145

а в формате JSON или HTML, то разница в размере была бы гораздо больше.

В процентном соотношении преимущество в производительности REST над SOAP, как видно из табл. 3, составляет 7,29% для записи и 9,17% для чтения. Учитывая, что разработанные приложения достаточно просты с точки зрения вычислений и не требуют больших производительных затрат, можно сделать вывод, что эффективность REST будет только возрастать в более ресурсоемких приложениях. Например, в таких, где происходит соединение с базой данных.

На эффективность работы приложения помимо количества передаваемых объектов и выбранного протокола также большое влияние оказывают такие факторы, как помехозащищенность соединения и скорость передачи данных [4]. Из-за этого скорость выполнения запросов к облаку может различаться при работе в разных сетях.

Заключение

В данной работе изучены такие технологии доступа к облачным сервисам, как SOAP и REST, проведено их сравнение с теоретической точки зрения, а также написаны приложения с идентичной функциональностью, позволяющие оценить производительность данных протоколов.

Анализ результатов показал, что REST эффективней SOAP за счет простоты и возможности выбора формата передачи данных.

Библиографический список

1. Лозовюк А. XML-RPC, вызов удаленных процедур с помощью XML / А. Лозовюк. – URL : http://citforum.ru/internet/xml/xml_rpc.

2. Лошин П. Протокол SOAP / П. Лошин // ComputerWorld Россия. – 2000. – № 35.

3. Хабрахабр. RESTandSOAP. Ч. 1. Почувствуйте разницу. – URL : http://habrahabr.ru/post/131343 (дата обращения 27.10.2011).

4. Яковлев В.В. Оценка влияния помех на производительность протоколов канального уровня / В.В. Яковлев, Ф.И. Кушназаров // Изв. ПГУПС. – 2015. – Вып. 1 (42). – С. 133–138.

5. Roy Fielding about HTTP, REST, WebDAV, JSR 170, and Waka. – URL : http://jonudell.net/udell/2006-08-25-a-conversation-with-roy-fielding-about-http-rest-webdav-jsr-170-and-waka.html (дата обращения 25.08.2006).

6. Roy Fielding on Versioning, Hypermedia, and REST. – URL : http://www.infoq.com/articles/roy-fielding-on-versioning.

7. W3.org. Официальный сайт Консорциума Всемирной Паутины. Спецификация SOAP версия 1.2. – URL : http://www.w3.org/2002/07/soap-translation/russian/part0.html.

References

1. Lozovyuk A. XML-RPC, vyzov udalennykh protsedur s pomoshchyu XML [XML-RPC, Remote Procedure Call by XML], available at: citforum.ru/internet/xml/xml_rpc.

2. Loshin P. *ComputerWorld Rossiya – ComputerWorld Russia*, 2000, no. 35.

3. Khabrakhabr. RESTandSOAP. Ch. 1. Pochustvuyte raznitsu [RESTandSOAP. Pt. 1. Feel the Difference], available at: habrahabr.ru/post/131343.

4. Yakovlev V. V. & Kushnazarov F. I. *Izvestiya PGUPS – Proc. Petersburg Transp. Univ*., 2015, Is. 1 (42), pp. 133-138.

5. Roy Fielding about HTTP, REST, WebDAV, JSR 170, and Waka, available at: jonudell.net/udell/2006-08-25-a-conversation-with-roy-fielding-about-http-rest-webdav-jsr-170-and-waka.html.

6. Roy Fielding on Versioning, Hypermedia, and REST, available at: infoq.com/articles/roy-fielding-on-versioning.

7. W3.org. Spetsifikatsiya SOAP versiya 1.2 [SOAP Specification Version 1.2], available at: w3.org/2002/07/soap-translation/russian/part0.html.

КУШНАЗАРОВ Фаррух Исакулович – аспирант, k.farruh@bk.ru; ЯКОВЛЕВ Валентин Васильевич – д-р техн. наук, профессор, jakovlev@pgups.ru; *ТУРДИЕВ Одилжан Акрамович – магистр, odiljan.turdiev@mail.ru (Петербургский государственный университет путей сообщения Императора Александра I).