УДК 004.7

ИСПОЛЬЗОВАНИЕ ПРОТОКОЛА SOAP В РАСПРЕДЕЛЕННЫХ ПРИЛОЖЕНИЯХ

Турашов А.С., Чередникова О.Ю.
Донецкий национальный технический университет

Со времени начала бурного развития сетевых технологий в конце 80-х годов разработчики стремились вывести методы программирования за пределы одного компьютера. Так, вначале программы взаимодействовали друг с другом по кабельному соединению через параллельный или последовательный порт, затем по сети, используя сетевые протоколы. Позднее были реализованы различные варианты удаленного вызова процедур (RPC, Remote Procedure Call), что давало возможности разработчику ПО не беспокоиться о том, как информация между модулями программы передается по сети. Потом на базе RPC создали протокол ORPC, в котором используются технологии DCOM, CORBA/IIOP или RMI (Remote Method Invocation). ORPC позволяет организовать взаимодействие по сети уже на уровне объектов. Дальнейшим развитием сетевых технологий можно считать появление протокола SOAP. [1]

Simple Object Access Protocol (SOAP) — это протокол на базе языка XML, который определяет правила передачи сообщений по Internet между различными прикладными системами. В основном он используется для удаленного вызова процедур. Изначально протокол SOAP разрабатывался с тем расчетом, что он будет функционировать «над» HTTP (дабы упростить интеграцию SOAP в Web-приложения), однако теперь могут быть задействованы и иные транспортные протоколы, например SMTP.

В распределенных системах SOAP используется для обеспечения взаимодействия разных уровней. На сегодняшний день существует множество технологий и протоколов, позволяющих без труда соединять элементы распределенных систем между собой. Одна из наиболее известных технологий – DCOM, позволяющая эффективно осуществлять RPC-вызовы, передавать и принимать данные, распределять нагрузку между несколькими back-end серверами. Однако у систем, построенных на DCOM, есть очень важный недостаток, затрудняющий взаимодействие уровня представления и уровня бизнес-логики через Internet. Хотя DCOM-приложения могут использовать TCP/IP для передачи вызовов RPC, большинство современных сетевых экранов будут запрещать передачу таких пакетов из соображений безопасности. Конечно, с помощью утилиты DCOMCNFG можно настроить DCOM на использование любого порта в диапазоне от 1024 до 65535. Но при изменении настроек одного из промежуточных файрволлов DCOM может перестать работать. Можно сказать, что DCOM является доминирующей технологией для обмена информацией и передачи вызовов в пределах корпоративной локальной сети, но при выходе за ее пределы DCOM приносит большое количество хлопот, связанных с настройкой портов, файрволлов и т.д. Для избежания длительной настройки системы был разработан протокол SOAP.

Протокол SOAP можно считать простым: данные, например имена вызываемых методов и параметры упаковываются в XML-документ (пакет SOAP), отсылаются на сервер, который возвращает ответ – также пакет SOAP содержащий результат выполнения процедуры. В роли сервера часто выступает ASP-страница. Способность пакетов SOAP преодолевать файрволлы и другие виды защиты с помощью HTTP составляет одновременно и преимущества и проблему безопасности. [2]

Протокол SOAP не является объектной архитектурой или платформой. Это простой протокол, который не может заменить таких технологий как Java/RMI, DCOM

или CORBA. Протокол SOAP дополняет данные протоколы. В спецификациях SOAP не описано решений защиты и самого доступа к объектам.
SOAP описывает преобразование в XML следующих элементов вызова:

- Запрос (SOAP Request).
- Отклик (SOAP Response).
- Сообщение об ошибке (SOAP Fault).

SOAP Request
Вызов метода по протоколу SOAP преобразуется в XML следующим образом:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope ...>
  <SOAP-ENV:Body ...>
  <SOAPSDK4:Add ...>
    <x>1</x>
    <y>2</y>
  </SOAPSDK4:Add>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

В приведенном примере запроса вызывается метод Add, которому передаются два параметра: 1 и 2. Информация о том, какой метод и у какого объекта необходимо вызвать, передается в заголовке. В случае протокола HTTP заголовок может выглядеть так:

```
<HTTPHeaders>
  <soapaction>"http://tempuri.org/Sample1/action/Adder.Add"</soapaction>
  <content-type>text/xml; charset="UTF-8"</content-type>
  <user-agent>SOAP Toolkit 3.0</user-agent>
  <host>ivan:8080</host>
  <content-length>516</content-length>
  <connection>Keep-Alive</connection>
  <cache-control>no-cache</cache-control>
  <pragma>no-cache</pragma>
</HTTPHeaders>
```

В теге SoapAction указывается, какое действие необходимо выполнить на сервере.

SOAP Response
Ответ сервера содержит значения возвращаемых параметров.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope ...>
  <SOAP-ENV:Body ...>
  <SOAPSDK4:AddResponse ...>
    <Result>3</Result>
  </SOAPSDK4:AddResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

В данном случае сервер ответил на вызов метода Add с параметрами 1 и 2 значением 3.

SOAP Fault

В случае возникновения ошибок до или во время обработки запроса сервер возвращает информацию об ошибке. Ошибки могут быть двух типов:

- Клиентские ошибки. Неправильный запрос к серверу. Запрос содержит неверные значения параметров, неправильно сформирован и т.д.
- Серверные ошибки. Ошибки могут быть связаны как с функционированием самого сервера (закончились свободные ресурсы), так и с обработкой запроса (компонент вернул ошибку).

Пример серверной ошибки:

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<SOAP-ENV:Envelope ...>
  <SOAP-ENV:Body ...>
  <SOAP-ENV:Fault>
    <faultcode>SOAP-ENV:Server</faultcode>
    <faultstring>Can add no more numbers</faultstring>
    <faultactor>http://ivan:8080/Sample1/Sample1.ASP</faultactor>
    <detail>
    <mserror:errorInfo ...>
      <mserror:returnCode>-2147467259 : Unspecified error
      </mserror:returnCode>
      <mserror:serverErrorInfo>
      <mserror:description>Can add no more numbers</mserror:description>
      <mserror:source>Sample1.Adder.1</mserror:source>
      </mserror:serverErrorInfo>
      <mserror:callStack>
      <mserror:callElement>
        <mserror:component>WSDLOperation</mserror:component>
        <mserror:description>Executing method Add failed
        </mserror:description>
        <mserror:returnCode>-2147352567 : Exception occurred.
        </mserror:returnCode>
      </mserror:callElement>
      ...
      </mserror:callStack>
    </mserror:errorInfo>
    </detail>
  </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

В теге **faultcode** указывается источник ошибки – клиент или сервер. **Faultstring** содержит строковое описание ошибки. **Faultfactor** – указывает URL, по которому обращался клиент. Тег detail содержит дополнительную информацию об ошибке (например, call stack).[3]

Использование протокола SOAP в распределенных приложениях

Наиболее эффективно использовать протокол SOAP в распределенных приложениях. Данный протокол позволяет создать распределенное приложение не на основе многопроцессорности и использования мощных серверов, а на основе распределения всех выполняемых приложением функций по локальным компьютерам сети Internet. Локальные компьютеры в данном случае являются серверами. Процесс взаимодействия клиента с сервером представлен на рисунке 1.[1]

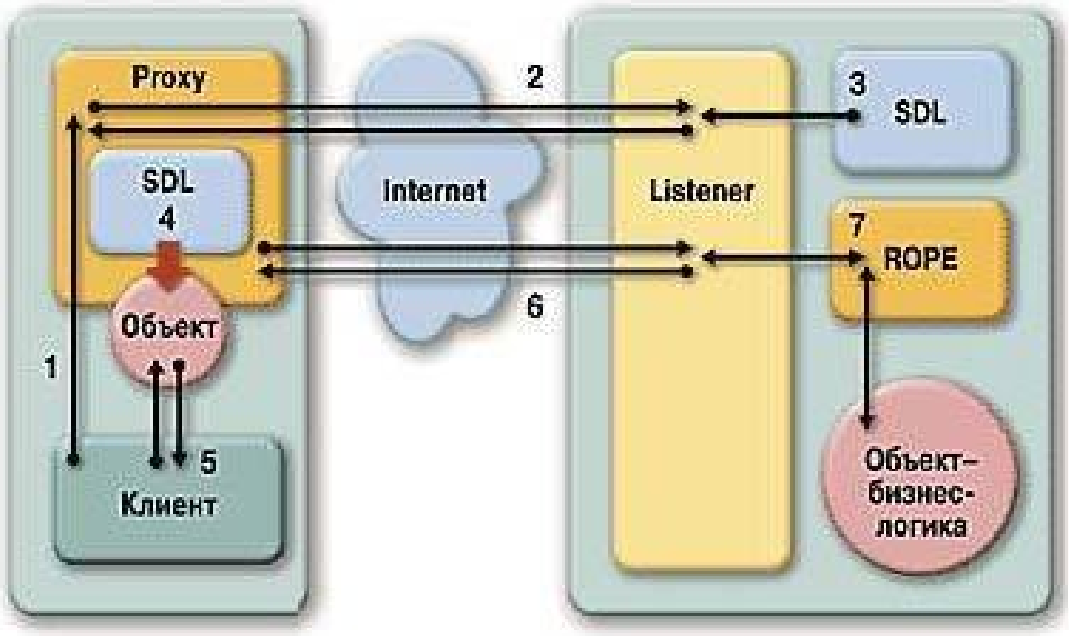

Рисунок 1 – Схема взаимодействия клиента и сервера по протоколу SOAP

Процедура реализации решения на основе SOAP следующая:

1. На клиенте нужно создать объект ROPE.Proxy (объект Proxy библиотеки ROPE).
2. С применением объектной модели библиотеки ROPE (Remote Object Proxy Engine) следует запросить с сервера SDL-файл с информацией о сервере, на основе которой библиотека автоматически реализует объект-заглушку, методы и свойства которого описаны в SDL-файле.
3. Когда клиент примет ссылку на полученный объект, то можно вызывать его методы и получать сведения о свойствах. Всякий вызов будет передаваться с помощью библиотеки ROPE и протокола SOAP на сервер, где любой SOAP-запрос встретит модуль Listener (Прослушиватель), определяющий его тип и должным образом реагирующий.
4. Если происходит запрос SDL-файла, то возвращается такой файл, а если вызов метода или свойства объекта, то Listener выделяет его из пришедшего XML-файла, выполняет запрос к расположенному на сервере COM-объекту, получает результат, используя ROPE, «запаковывает» его в XML и отсылает обратно.
5. Клиент получает файл и может проводить дальнейший анализ.[3]

Литература

[1] XML и SOAP: программирование для серверов BizTalk. Новейшие технологии/ Пер. с англ, М.: Издательско-торговый дом “Русская Редакция”, 2001. – 496 стр.

[2] Разработка Web-сервисов XML и серверных компонентов на Microsoft Visual Basic .NET и Microsoft Visual C# .NET. Учебный курс MCAD/MCSD/Пер. с англ. — М.: Издательско-торговый дом «Русская Редакция», 2004. -576 стр.: ил.

[3] RSDN Magazine #1 – 2003, М., Роспечать, 2003