Вестник Сыктывкарского университета.
Сер.1. Вып.6. 2006

УДК 519.717

ВЕБ–СЕРВИС ДЛЯ КОМБИНИРОВАННОГО АЛГОРИТМА ЦЕЛОЧИСЛЕННОЙ ЗАДАЧИ ЛИНЕЙНОГО РАСКРОЯ

В.Л. Никитенков, В.И. Ясинский

Рассмотрены проблемы, возникающие при работе по сети и Интернет, превосходство веб–сервисов над другими серверными приложениями, краткое описание реализованного веб–сервиса, примеры вызова веб–сервиса.

Сеть Интернет стала общепризнанным фактором деловой и общественной жизни. Широкая распространенность и возросшая пропускная способность создают условия, при которых выгодно решать многие задачи при помощи интернет–технологий. Однако Интернет объединяет в себе много различных платформ, а информация содержится в разнообразных источниках данных. Поэтому актуальна проблема связи таких разнородных данных, а также создания способа, который позволяет получать их в виде удобном для дальнейшей обработки.

По требованию руководства ОАО "МБП–СЛПК" для комбинированного алгоритма решения целочисленной задачи линейного раскроя был реализован веб–сервис. А почему именно веб–сервис? Приложения и раньше обменивались информацией через Интернет — для этого служат DCOM-, CORBA-, Java RMI-, HTTP GET/POST-запросы. Все эти способы обладают одним или несколькими недостатками: у них отсутствует расширяемость, есть проблемы с доступом через сетевой экран, они привязаны к определенной платформе программирования, возникают сложности при разработке сервиса и клиентского приложения. Типичные примеры - различные финансовые серверы выдают курсы валют и котировки по-разному, та же неразбериха с погодой, статистикой и другими данными, что заметно усложняет разработку b2b–(business-to-business) систем и порождает сложные системы.

А что такое веб–сервис? Веб–сервис это универсальный метод для вызова удаленных процедур. Благодаря своей простоте, независимости от языков программирования и способности кросс–платформенного взаимодействия веб–сервисы имеют большие шансы стать основным протоколом для построения распределенных систем. Веб–сервисы преобразуют XML–документы (Extensible Markup Language, XML) в ИТ–системах. Веб–сервисы — это XML-приложения, осуществляющие связывание данных с программами, объектами, базами данных либо с деловыми операциями целиком. Между веб–сервисом и программой осуществляется обмен XML-документами, оформленными в виде сообщений. Стандарты веб–сервисов определяют формат таких сообщений, интерфейс, которому передается сообщение, правила привязки содержания сообщения к реализующему сервис приложению и обратно, а также механизмы публикации и поиска интерфейсов. Веб–сервисы могут использоваться во многих приложениях. Независимо от того, откуда запускаются веб–сервисы, с настольных компьютеров клиентов или с переносных, они могут использоваться для обращения к таким интернет–приложениям, как система предварительных заказов или контроля выполнения заказов. Веб-сервисы пригодны для B2B–интеграции (business-to-business), замыкая приложения, выполняемые различными организациями, в один производственный процесс. Веб–сервисы также могут решать более широкую проблему интеграции приложений предприятия (Enterprise Application Integration, EAI), осуществляя связь нескольких приложений одного предприятия с несколькими другими приложениями, размещенными как "до", так и "после" брандмауэра. Во всех перечисленных случаях технологии веб–сервисов являются "связующим звеном", объединяющим различные части программного обеспечения. Веб–сервисы представляют собой оболочку, обеспечивающую стандартный способ взаимодействия с прикладными программными средами, такими как системы управления базами данных (СУБД), .NET, J2EE (Java2 Platform, Enterprise Edition), CORBA (Common Object Request Broker Architecture), посредники пакетов планирования ресурсов предприятия (Enterprise Resource Planning, ERP), брокеров интеграции и пр. Интерфейсы веб–сервисов получают из сетевой среды стандартные XML–сообщения, преобразуют XML–данные в формат, "понимаемый" конкретной прикладной программной системой, и отправляют ответное сообщение (последнее — не обязательно). Программная реализация веб–сервисов (базовое программное обеспечение, нижний уровень) может быть создана на любом языке программирования с использованием любой операционной системы и любого связую-

щего программного обеспечения (middleware). Веб–сервисы объединяют программирование и концепции Сети. Веб–сервисы сочетают параметры программных приложений и абстрактные характеристики Сети. Современные интернет–технологии частично достигают своих целей, поскольку они определены на очень высоком отвлеченном уровне, что обеспечивает совместимость с любой операционной системой, любым программным и аппаратным обеспечением. Инфраструктура, основанная на применении веб–сервисов, пользуется этим уровнем абстракции и включает в себя связанную с данными семантическую информацию, то есть веб-сервисы определяют не только данные, но и порядок обработки и преобразования этих данных в базовые программные приложения и обратно.

1. Описание веб–сервиса.

Решение задачи форматного раскроя с оптимизацией по материалам и количеству различных планов раскроя при наличии люфта в векторе требований на форматы реализовано в виде веб–сервиса, построенного на основе сессионного бина. Сессионый бин состоит из трех классов:
CPartTambourEJB_Luft - удаленный интерфейс,
CPartTambourEJB_LuftHome - домашний интерфейс,
CPartTambourEJB_LuftBean — класс бина.

В классе бина реализован алгоритм. Реализация состоит из двух классов: CPartTambourEJB_LuftBean - в нем производится оптимизация по отходам и класса main1 - в нем сам раскрой и оптимизация по числу различных раскладок.

В классе CPartTambourEJB_Luft - заголовок метода, который открыт для клиента (в данном случае это метод start). Веб–сервис собран в архив Для клиента открыт метод start. Этому методу передаются исходные данные в следующем порядке:

Smax — максимальная ширина тамбура(мм)

Smin — минимальная ширина тамбура(мм)

MaxEdge — максимальная кромка(мм)

MinEdge — минимальная кромка(мм)

KnifeInterval — минимальное расстояние между ножами(мм)

KnifeCount — количество ножей на ПРС

MinInterval — минимальный попутчик(мм)

String1 — строка, содержащая форматы через пробел

String2 — строка, содержащая максимальное количество каждого формата через пробел

String3 — строка, содержащая разность между максимальным и минимальным количеством каждого формата

Результат:

String4="Sets=N Total_ostatok=N1 Efficiency=Eff % ! $\mathbf{A_i} * [\mathbf{X_i} * \mathbf{C_i} \mathbf{XX_i} * \mathbf{CC_i} \ldots \| \mathbf{B_i} \| \overline{\mathbf{D}}_\mathbf{i} \| \mathbf{eff_i} \%]$ ...",

$i = [1..q]$, где q — количество различных съёмов,

Sets=N — количество съёмов

S=Smax-2*MinEdge

$\mathbf{A_i}$ — количество съёмов, нарезаемых i-ым способом

$\mathbf{X_i}$ — количество форматов $\mathbf{C_i}$ в i-ом способе и т.д.

$\mathbf{B_i}$ — используемая ширина тамбура на i-ом съёме

$\mathbf{D_i}$ — остаток на i-ом съёме $\mathbf{D_i} = \mathbf{S} - \mathbf{B_i}$

$\mathbf{eff_i}$ — эффективность на i-ом съёме $\mathbf{eff_i} = (\mathbf{B_i}/\mathbf{S}) * \mathbf{100}\%$

Total_ostatok=N1 — общий остаток(**Total_ostatok=**$\sum_{\mathbf{i=1}}^{\mathbf{q}} \mathbf{Di}$)

Efficiency=Eff% — эффективность по материалам(**Efficiency=**(1-Total_ostatok/(S*Sets)))*100%

Пример:

Форматы	Мин	Макс	Разность
600	26	28	2
816	17	19	2
840	15	16	1
888	64	66	2
1440	106	113	7
1490	31	166	135
1550	23	25	2

Smax=8480
Smin=8460
MaxEdge=100
MinEdge=20
KnifeInterval=200
KnifeCount=18
MinInterval=170
String1="600 816 840 888 1440 1490 1550"
String2="28 19 16 66 113 166 25"
String3="2 2 1 2 7 135 2"

String4="Sets=40 Total_ostatok=424 Efficiency=99.87% ! 7*[2*1550 1*1490 1*1440 4*600 ||8430 ||10 ||99.88%] 3*[4*1490 1*840 2*816 ||8432 ||8 ||99.91%] 20*[4*1440 3*888 ||8424 ||16 ||99.81%] 6*[1*1550 1*1490 2*1440 1*888 2*816 ||8440 ||0 ||100.0%] 3*[1*1550 1*1490 2*1440 3*840 ||8440 ||0 ||100.0%] 1*[3*1490 1*1440 3*840 ||8430 ||10 ||99.88%]"

Для того, чтоб вызвать веб–сервис, необходимо, чтобы он был развернут на Weblogic server-е. Далее, просто нужно узнать ссылку на WSDL файл, в котором находится описание сервиса. WSDL файл это документ в формате XML, описывающий методы, предоставляемые веб–сервисом. Также параметры методов, их типы, названия и местонахождение Listener'а сервиса. SOAP Toolkit визард автоматически генерирует этот документ.

Вот как выглядит WSDL файл:

<?xml version="1.0"encoding="UTF-8"?>
- <definitions xmlns:tns="**http://cparttambour_with_luft1**"
xmlns:wsr="**http://www.openuri.org/2002/10/soap/reliability/**"
xmlns:mime="**http://schemas.xmlsoap.org/wsdl/mime/**"
xmlns:soap12="**http://schemas.xmlsoap.org/wsdl/soap12/**"
xmlns:http="**http://schemas.xmlsoap.org/wsdl/http/**"

```
      xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:soap12enc="http://www.w3.org/2003/05/soap-encoding"
      xmlns:conv="http://www.openuri.org/2002/04/wsdl/conversation/"
      xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
      xmlns:s="http://www.w3.org/2001/XMLSchema"
      xmlns="http://schemas.xmlsoap.org/wsdl/"
     targetNamespace="http://cparttambour_with_luft1" >
  <message name="getSample"/>
- <message name="getSampleResponse" >
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
    "partns:string"name="result"/>
      </message>
- <message name="start" >
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:int"name="intVal"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:int"name="intVal0"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:int"name="intVal1"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:int"name="intVal2"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:int"name="intVal3"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:int"name="intVal4"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:int"name="intVal5"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:string"name="string"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:string"name="string0"/>
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:string"name="string1"/>
      </message>
- <message name="startResponse" >
  <part xmlns:partns="http://www.w3.org/2001/XMLSchema"type=
  "partns:string"name="result"/>
      </message>
- <portType name="CPartTambourBean1Port" >
- <operation name="getSample" >
  <input message="tns:getSample"/>
  <output message="tns:getSampleResponse"/>
      </operation>
- <operation name="start" >
  <input message="tns:start"/>
  <output message="tns:startResponse"/>
      </operation>
      </portType>
```

```
- <binding type="tns:CPartTambourBean1Port"name=
  "CPartTambourBean1Port" >
    <soap:binding style="rpc"transport=
    "http://schemas.xmlsoap.org/soap/http"/>
- <operation name="getSample" >
    <soap:operation style="rpc"soapAction= />
- <input>
    <soap:body namespace="http://cparttambour_with_luft1"
      encodingStyle ="http://schemas.xmlsoap.org/soap/encoding/"
       use="encoded"/>
       </input>
- <output>
    <soap:body namespace="http://cparttambour_with_luft1"
       encodingStyle= "http://schemas.xmlsoap.org/soap/encoding/"
      use="encoded"/>
      </output>
       </operation>
- <operation name="start" >
    <soap:operation style="rpc"soapAction= />
- <input>
    <soap:body namespace="http://cparttambour_with_luft1"
       encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
       use="encoded"/>
       </input>
- <output>
    <soap:body namespace="http://cparttambour_with_luft1"
       encodingStyle ="http://schemas.xmlsoap.org/soap/encoding/"
       use="encoded"/>
       </output>
       </operation>
       </binding>
- <service name="CPartTambourBean1" >
- <port name="CPartTambourBean1Port"binding=
  "tns:CPartTambourBean1Port" >
       <soap:address location= "http://syk-sv30:7001/CPartTambour
       -web-services/CPartTambourBean1"/>
       </port>
       </service>
       </definitions>
```

Затем пишется код клиента. Это можно сделать несколькими способами.

Приведем два примера:

1.

```
import package.CPartTambourBean1;
...
String wsdlUrl = "http://syk-1609:7001/CPartTambour-web-services/
```

```
CPartTambourBean1?WSDL";
CPartTambourBean1 service=new CPartTambourBean1_Impl( wsdlUrl );
CPartTambourBean1Port port=service.getCPartTambourBean1Port();
result=port.start( ... );
```

Это пример вызова сервиса, разработанного с помощью инструмента JBuilder фирмы Borland, развернутого на syk-1609.

2.

```
package wsconnect;
/**
* @author YasinskyV
*/
import java.util.*;
import net.n3.nanoxml.*;
import wsdynamicclient.*;
import org.apache.axis.encoding.SerializationContext.*;

public class Main {
     private static String wsdl="http://syk-sv18:7003/CPartTambour-web-
services/CPartTambourBean1?WSDL";
public static void main(String[] args) {
    try{
          wsdynamicclient.BC_invoker_xml invoker=new
          wsdynamicclient.BC_invoker_xml(wdl);
          String[ ] wsargs = new String[1];
          String result = ;
String params = " <params SSmax = '8440' SSmin = '0' MMaxEdge = '0'
MMinEdge = '0' KKnifeInterval = '0' KKnifeCount = '18' MMinInterval='0'
cf = '600 800 400' Lmax = '53 54 55' difference = '1 10 20' /> ";
          wsargs[0]=params;
          if (invoker != null) result = invoker.invokeMethod ("start001", null,
          wsargs);
          System.out.println(result);
      }catch(Exception e){e.printStackTrace( );}
}
}
```

Это вызов сервиса, развернутого на syk–sv18. Там для клиента открыт метод start001, которому на вход подается строка вида
" <params SSmax = '8480' SSmin = '8460' MMaxEdge = '100' MMinEdge = '20' KKnifeInterval = '200' KKnifeCount = '18' MMinInterval='200' cf

= ’600’ Lmax = ’53’ difference = ’1’ />"
cf – эквивалентно String1 из примера
Lmax – String2
difference – String3
Результат возвращается в том же виде.

Макс. ширина тамбура (мм) =	8440
Мин. ширина тамбура (мм) =	0
Макс. кромка (мм)=	0
Мин. кромка (мм)=	1
Мин. раст. между ножами=	1
Кол-во ножей на ПРС =	18
Мин. попутчик=	1
Строка содержащая форматы через пробел =	600 800 400
Строка содержащая макс. количество каждого формата через пробел (в рулонах) =	53 54 55
Строка содержащая разность между макс. и мин. кол-м каждого формата(в рулонах) =	1 10 20

STROKA RES==Sets=10 Total_ostatok=380 Efficiency=

99.55% ! 4*[10*600*400 ||8400 || 38 || 99.55%] 6*[8*800 2*600 2*400 || 8400 || 38 ||99.55%]

SUBMIT

общее кол-во съемов = 10
общий остаток = 380
эффективность по материалам= 99.55%

Номер комбинации ножей	Кол-во съемов	Кол-во	Формат	Использ. ширина	Остаток на съеме	Эффект-сть%
1	4	10	600	8400	38	99.55%
		6	400			
2	6	8	800	8400	38	99.55%
		2	600			
		2	400			

Также создан пользовательский интерфейс для сессионного бина в виде веб–приложения diagrams развернутого на syk-sv30. К нему можно обратиться по ссылке http://syk-sv30:7001/diagrams/material.jsp?times_sess=0

Литература

1. **Никитенков В.Л., Саковнич Д.Ю.** Реализация комбинированного алгоритма решения целочисленной задачи линейного раскроя. *Вестник Сыктывкарск. ун-та. Сер.1: Мат. Мех. Инф. 2006. Вып.6. С. 199-209.*

2. **Хабибулин Э.Ш.** Самоучитель Java2. СПб: БХВ–Петербург. 2005.

3. **Эккель Б.** Философия Java. Библиотека программиста. СПб: Питер, 2003.

4. **Ньюкомер Э.** Веб–сервисы. XML, WSDL, SOAP и UDDI. Для профессионалов. СПб: Питер, 2004.

5. BEA WebLogic Server®Programming Web Services for WebLogic Server Version 9.0 Revised: July 22, 2005.

Summary

Nikitenkov V.L.,Yasinsky V.I. Web–services of complex algorithm for integer–valued problem of linear cutting

The problems of working over a network and the Internet, the superiority of Web–services over the other server–applications, a short description of the applied Web–service and the examples of Web–service query are discussed.

Сыктывкарский университет *Поступила 19.04.2006*