Gerador de Web Services para cadeias de tranformações de documentos XML

José Carlos Ramalho, Pedro Taveira, Ricardo Ferreira e Vasco Rocha

DI/UM

jcr@di.uminho.pt pjstaveira@netcabo.pt ricardomiguel@myrealbox.com
vmrocha@netcabo.pt

Resumo Os Web Services são cada vez mais utilizados na implementação de aplicações distribuídas. Apesar de utilizarem normas standard de baixa complexidade, a quantidade de normas utilizadas e a interacção entre elas aumenta razoavelmente a complexidade da sua implementação. A ideia inicial deste projecto foi estudar a viabilidade da geração automática de Web Services partindo de uma especificação abstracta do processo. Para isso definiu-se uma linguagem XML muito simples com a qual se especifica um processo ou uma cadeia de processos e criaram-se as transformações necessárias para desta especificação se chegar ao Web Service final.
Posteriormente, decidiu-se particularizar o sistema para Web Services que implementam cadeias de transformação XML. Estas cadeias de transformação correspondem a aplicações XML de segunda geração ou, por outras palavras, com reflexão a nível da transformação.
Neste momento, o sistema possui uma interface em .Net com a qual é possível especificar a cadeia de operações a realizar pelo Web Service e que permite gerar a aplicação servidor do serviço e a aplicação cliente, ambas em tecnologia .Net.

1 Introdução

Um dos principais problemas no mundo informático é a interoperabilidade estre as diversas plataformas. O aparecimento dos Web Services veio atenuar este factor, utilizando normas standard como o XML e tornando possível a comunicação entre aplicações como se estas fossem caixas negras, ou seja, é possível comunicar com as aplicações de uma forma normalizada desconhecendo os detalhes da sua implementação.

Mesmo com estas vantagens, à primeira vista, criar um Web Service pode não ser muito simples. Há várias camadas de software envolvidas. Há um conjunto de normas que se inter-relacionam e que o programador deve conhecer. Por estas razões, surgiram no mercado várias plataformas para desenvolvimento de Web Services em várias linguagens de programação. Se se optar por uma destas plataformas a criação de servidores de serviços e de clientes pode ser sistematizada podendo mesmo ocultar-se todo o processo de implementação. Foi este pressuposto que deu o mote deste trabalho. Assim o primeiro objectivo foi criar uma

linguagem XML com a qual fosse possível especificar ao nível mais abstracto possível um Web Service, e desenvolver as ferramentas necessárias que a partir da especificação de um Web Service gerasse o código para o implementar, quer o servidor quer o cliente.

Cedo se verificou que só se conseguia uma verdadeira abstracção se se concretizásse um pouco o Web Service, i. e., se o sistema que se estava a desenvolver estivesse preparado para gerar Web Services de uma determinada espécie. Com esse objectivo, seleccionou-se um conjunto de aplicações para as quais já existia há algum tempo a intenção de as expôr na Web. Estas aplicações são, muitas vezes, designadas por transformações XML de ordem superior e caracterizam-se por serem transformações com reflexão numa determinada fase ou em várias fases da sua execução. A execução de uma aplicação destas não é simples para o utilizador comum e há muito que se pensava em expô-las na Web como um serviço ocultando os pormenores, às vezes complicados da sua execução.

A plataforma escolhida para o desenvolvimento deste projecto foi a plataforma .NET da Microsoft. A escolha desta plataforma para o desenvolvimento deste projecto deveu-se ao facto de permitir uma implementação relativamente simples e intuitiva de Web Services.

Neste artigo, descrevem-se os passos seguidos na implementação de um gerador de Web Services para o tipo de aplicações descrito acima. O sistema desenvolvido pode ser adaptado facilmente para outro tipo de aplicações.

Na próxima secção descreve-se com um pouco mais de detalhe as cadeias de transformações de documentos XML. Na secção seguinte, apresenta-se a linguagem XML que se definiu para especificar a cadeia de processos que o Web Service a gerar deve implementar. A seguir descreve-se um pouco o processo de geração propriamente dito. O artigo termina com uma secção onde se faz uma síntese do trabalho realizado e onde se apontam linhas para trabalho futuro.

2 Cadeias de Transformações de Documentos XML

As aplicações XML implementadas através de cadeias de transformações com reflexão são aquilo que às vezes se designa por aplicações de ordem superior. Neste tipo de aplicações, o utilizador trabalha numa camada de abstracção superior ao de uma aplicação normal. A figura 1 apresenta o esquema duma aplicação deste tipo.

Como se pode ver na figura 1 o utilizador produz um documento XML com a especificação abstracta de um determinado processo de transformação; este documento é processado por uma metastylesheet (que implementa o nível superior de abstracção) e que gera como resultado uma stylesheet XSL específica que implementa o processo especificado; esta nova stylesheet pode depois ser usada para aplicar o processo de transformação especificado no nível superior a vários documentos XML concretos.

A figura 2 representa o mesmo esquema genérico mas agora concretizado numa aplicação concreta que utiliza estes dois níveis e que foi designada pelo autor por *XPW: Xpath Wrapper, uma ferramenta para auxiliar no ensino de*

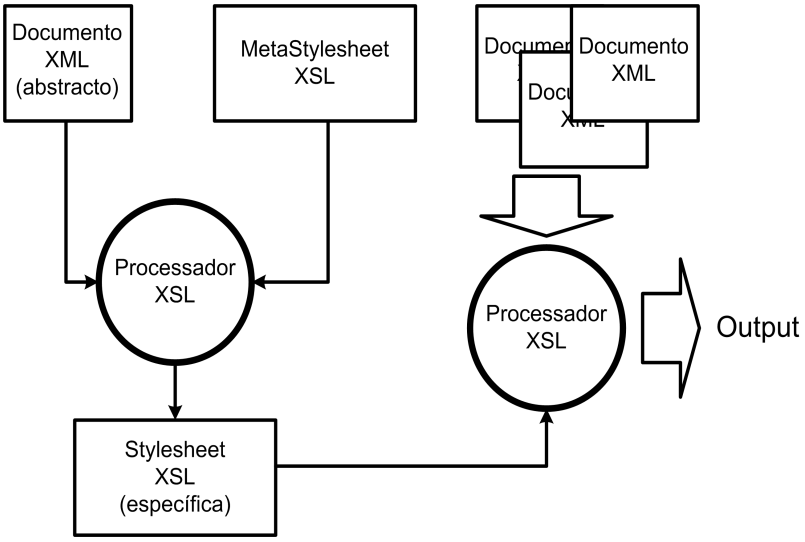

Figura 1. Esquema genérico de uma transformação com reflexão

XPath. Como o próprio nome indica esta ferramenta tem vindo a ser utilizada no ensino de XPath.

O XPW foi construído de modo a permitir ao utilizador realizar queries, especificadas em XPath, ao conteúdo de documentos XML. Assim, numa primeira etapa, o utilizador especifica num documento XML as queries que quer realizar; numa segunda etapa, esse documento é passado a um processador de XSL conjuntamente com a metastylesheet que implementa o nível abstracto do XPW; deste processo resulta uma stylesheet que implementa as queries especificadas; quando aplicada a um documento XML esta stylesheet irá extrair as partes seleccionáveis com as queries especificadas.

Em resumo, uma aplicação deste tipo tem dois níveis de transformação. O primeiro do abstracto para o concreto, e o segundo a transformação em concreto. Se, por exemplo, se acrescentasse ao XPW uma terceira transformação para tratar os resultados teríamos uma cadeia de três transformações.

Quando já se trabalha há algum tempo na área da transformação documental (documentos XML) é normal que o nível de abstracção, das aplicações que se vão criando, vá aumentando. Isto vai-se traduzindo em, cada vez maiores, cadeias de transformações.

É assim que surge a ideia de tornar estas cadeias transparentes para o utilizador. A estratégia para as ocultar, que foi seguida neste trabalho, foi colocá-las por detrás de um Web Service. Como este Web Service é, em termos funcionais, muito semelhante de umas aplicações para as outras, surgiu também a ideia de tornar a sua geração automática a partir da especificação abstracta de uma cadeia de transformações.

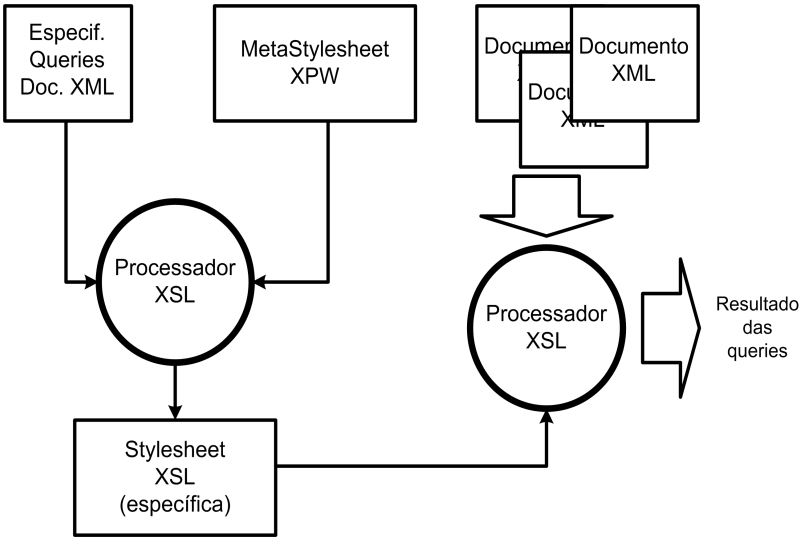

Figura 2. Esquema do XPath Wrapper

Na próxima secção, apresenta-se a linguagem XML definida para especificar cadeias de transformação.

3 Especificação de Cadeias de Transformação

O primeiro passo no processo de geração de Web Services para cadeias de transformações de documentos XML é a definição da cadeia de transformação. Para isso foi definida uma linguagem XML.

Para especificar uma cadeia de transformações começa-se por indicar todos os documentos XML e XSL que vão participar nas transformações da cadeia. Cada documento terá um identificador único e um atributo que identifica a sua localização (cliente – o documento é fornecido pelo cliente; servidor – o documento encontra-se no servidor).

Depois de definidos todos os documentos XML e XSL é preciso definir definir as transformações que precisam de ser executadas. Uma transformação define-se indicando qual o documento XML e qual a stylesheet XSL que irão participar nela. O documento e a stylesheet são referidos pelos identificadores únicos associados aos documentos aquando a sua definição. Assim, uma transformação é definida por um par (id_1, id_2): o atributo id_1 identifica o documento a ser transformado e o atributo id_2 identifica a folha de estilo a aplicar. Cada transformação também tem um identificador único que ficará depois associado ao seu resultado. Desta maneira, o resultado de uma transformação poderá ser utilizado como entrada de uma nova transformação.

O utilizador também deve associar um título e um nome à cadeia. O título irá aparecer na página do cliente. O nome possui um atributo que permite especifiar

se o resultado da cadeia de transformações é HTML ou XML. Esta funcionalidade foi implementada porque muitas vezes, da última transformação da cadeia resulta uma página HTML para melhor se poder visualizar os resultados obtidos e os Web Services precisam de indicar que tipo de mensagens estão a trocar.

Na figura 3 mostra-se uma representação gráfica da estrutura da linguagem.

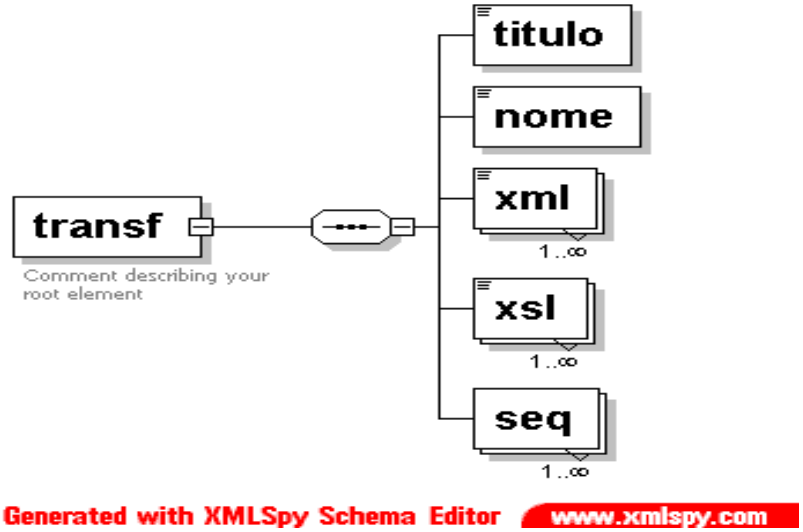

Figura 3. Schema da Linguagem de Anotação para Especificação de Cadeias de Tranformações

A título de exemplo, apresenta-se a seguir, a especificação da cadeia de transformações para uma utilização do XPW.

```
<?xml version="1.0" encoding="UTF-8"?>
<transf>
  <titulo>Web Service do XPath</titulo>
  <nome resp="xml">Queries XPath</nome>
  <xml id="poema" tipo="cliente">Ficheiro do Poema</xml>
  <xml id="querie" tipo="cliente">Ficheiro com as Queries</xml>
  <xsl id="xpath" tipo="servidor">Ficheiro com a metastylesheet</xsl>
  <seq id1="querie" id2="xpath" id="res"/>
  <seq id1="poema" id2="res" id="fim"/>
</transf>
```

A cadeia de transformações descrita acima define uma cadeia onde estão envolvidos dois documentos XML (`poema` e `querie`, localizados no cliente) e um documento XSL (`XPW` localizado no servidor). O documento `querie` é transformado pela folha de estilo `XPW` e o seu resultado irá ser uma folha de estilo que irá transformar o documento poema, obtendo-se assim o documento final.

Na próxima secção, discute-se a geração do serviço.

4 Geração do Web Service

A parte crítica deste projecto é a geração do Web Service a partir da linguagem descrita na secção anterior. No sentido de simplificar a comunicação entre o cliente e o servidor, decidiu-se utilizar `Strings` para representar os documentos que são transportados pela "rede". Por este motivo, a implementação de um cliente tornou-se quase obrigatória dado que, para podermos utilizar a página de teste que a plataforma .NET disponibiliza teríamos que introduzir todo o documento à mão nas respectivas caixas de texto. A geração do código do cliente e do servidor é efectuada por duas folhas de estilo distintas, que aplicadas a um documento XML escrito na linguagem descrita na última secção, irão gerar duas vistas distintas: o código do cliente e o código do servidor.

As stylesheets são demasiado extensas para serem discutidas aqui. No entanto, apresenta-se a seguir o servidor gerado para a especificação exemplo mostrada acima.

4.1 Servidor Gerado

```
<%@ WebService Language="c#" Class="WSGen.Queries XPath" %>
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Web;
using System.Web.Services;
using System.Xml;
using System.Xml.XPath;
using System.Xml.Xsl;
using System.IO;
using System.Xml.Serialization;

namespace WSGen
{
    [WebService(Namespace="http://www.WSGen.com/")]
    public class Queries XPath : System.Web.Services.WebService
    {
        [WebMethod]
        public string transf(string poemap, string queriep)
        {
            XmlDocument poema = new XmlDocument();
            poema.LoadXml(poemap);
            XmlDocument querie = new XmlDocument();
            querie.LoadXml(queriep);
            XslTransform xpath = new XslTransform();
            XmlTextReader xpathr = new XmlTextReader(Server.MapPath("xpath.xslt"));
            xpathr.MoveToContent();
            xpath.Load(xpathr);
```

```
30              XmlReader resp = xpath.Transform(querie, null);
31              resp.MoveToContent();
32              XslTransform res = new XslTransform();
33              res.Load(resp);
34              XmlReader fimp = res.Transform(poema, null);
35              fimp.MoveToContent();
36              XmlDocument fimx = new XmlDocument();
37              fimx.Load(fimp);
38              return fimx.DocumentElement.OuterXml;
39          }
40      }
41  }
```

Note que o código gerado para o servidor é implementado na linguagem C# e o gerado para o cliente é em .ASP da plataforma .NET.

5 Conclusões

Para simplifiar ainda mais todo este processo criou-se um interface em C# que permite ao utilizador especificar a linguagem de uma forma intuitiva, bem como gerar o servidor e o respectivo cliente a partir das folhas de estilo de uma forma bastante simples e automática.

A utilização da aplicação desenvolvida no ensino começará já nos próximos meses. Utilizando esta plataforma pode-se criar Web Services que ocultam os pormenores mais complexos das aplicações fazendo com que o aluno se concentre apenas naquilo que o nível de abstracção mais elevado da aplicação lhe exige.