Web Service de Consulta de CEP e Rastreamento de Encomendas dos Correios
Exercício Computacional I - Sistemas Distribuídos

Rafael Gonçalves de Oliveira Viana[1]
9 de outubro de 2017

[1]Sistemas de Informação – Universidade Federal do Mato Grosso do Sul (UFMS)
Caixa Postal 79400-000 – Coxim – MS – Brazil

rafael.viana@aluno.ufms.br

Resumo. *Este relatório introduz a arquitetura de um Web Service assim como seus componentes, o mesmo relata como foi implementado um Web Server que possui uma página web e um serviço para busca de CEP e rastreamento de encomendas dos Correios, utilizando Angular no front-end e NodeJS no back-end.*

1. Introdução

Com a diseminação de dispositivos com conexeção a Internet na sociedade atual, houvesse a necessidade de criar/gerenciar serviços web *Web Services*, estes necessarios para atender as mais diversas funções como o rastreio de encomendas e consultas de CEPs entre outros serviços. Este relatório demonstrará a arquitetura de um *Web Service* e como foi implementado um *Web Service* de consulta de CEP e rastreamento de encomendas dos correios, juntamente com uma página web que consome os serviços. O código deste relatório esta em sua totalidade no endereço http://github.com/rafaelgov95/SD/Projeto-Correios-Angular2, e o mesmo pode ser reutilizado sobre a licença MIT.

2. Arquitetura de um Web Service

Para poder identificar qual arquitetura que um *Web Service* deve possuir devemos examinar os papéis individuais de cada ator no *Web Service* e examinar a pilha emergente de protocolo que o mesmo pretende utiliza. O mesmo pode ser publicado na intranet ou na Internet, provendo três possiveis formatos de serviços: Provedor de Serviço, Solicitante de Serviço e o Registro de Serviço. Para mais detalhes desta sessão incentivo consultar [1].

1. **Provedor de serviço:** Fornece serviços web. O provedor de serviços implementa o serviço e disponibiliza-o na Internet ou intranet.
2. **Solicitante de Serviço:** Este é um consumidor do serviço web. O solicitante utiliza um serviço da Web existente abrindo uma conexão de rede e enviando uma solicitação XML.
3. **Registro de serviço:** Este é um diretório de serviços logicamente centralizado. O registro fornece um lugar central onde os desenvolvedores podem publicar novos serviços ou encontrar os existentes. Ele serve como centro de compensação centralizado para empresas e seus serviços.

2.1. Pilha de protocolo de um *Web Service*

Uma segunda opção para identificar qual arquitetura o *Web Service* deve utiliza é examinar a pilha emergente de protocolo que pretende utilizar. Cada dia existe mais tipos de protocolos porém existem 4 tipo que se destacam: Serviço de Transporte - ”FTP”, Mensagens ”XML”, Descrição de Serviço ”WSDL”e Descoberta de Serviço ”UDDI”

1. **Serviço de transporte:** Esta camada é responsável pelo transporte de mensagens entre aplicativos. Atualmente, esta camada inclui o protocolo de transporte de hipertexto (HTTP), protocolo de transferência de correio simples (SMTP), protocolo de transferência de arquivos (FTP) e protocolos mais recentes, como o protocolo de intercâmbio extensível de blocos (BEEP).
2. **Mensagens XML:** Esta camada é responsável por codificar mensagens em um formato XML comum para que as mensagens possam ser entendidas em cada uma das extremidades. Atualmente, esta camada inclui XML-RPC e SOAP.
3. **Descrição do Serviço:** Esta camada é responsável por descrever a interface pública para um serviço web específico. Atualmente, a descrição do serviço é tratada através do Web Service Description Language (WSDL).
4. **Descoberta do serviço:** Esta camada é responsável por centralizar os serviços em um registro comum e fornecer funcionalidades fáceis de publicação / pesquisa. Atualmente, a descoberta do serviço é tratada através de Descrição Universal, Descoberta e Integração (UDDI).

2.2. Principais Serviço de Transporte de um Web Service

Os protocolos da camada inferior ficam respornsaveis pelo transporte de serviços. Essa camada é responsável por transportar como exemplo mensagens XML e JSON entre dois computadores.

1. **Hyper Text Transfer Protocol (HTTP):** É a opção mais popular para o transporte de serviços. O HTTP é simples, estável e amplamente implantado. Além disso, a maioria dos firewalls permitem o tráfego HTTP. Isso permite que mensagens REST ou SOAP se mostrem como mensagens HTTP. Isso é bom se você quiser integrar aplicativos remotos, mas eleva uma série de preocupações de segurança.
2. **Bloqueia o protocolo de troca extensível (BEEP)** Esta é uma alternativa promissora para o HTTP. O BEEP é uma nova estrutura da Task Force de Engenharia da Internet (IETF) para a construção de novos protocolos. O BEEP está em camadas diretamente no TCP e inclui uma série de recursos internos, incluindo um protocolo inicial de handshake, autenticação, segurança e tratamento de erros. Usando BEEP, pode-se criar novos protocolos para uma variedade de aplicações, incluindo mensagens instantâneas, transferência de arquivos, distribuição de conteúdo e gerenciamento de rede. O SOAP não está vinculado a nenhum protocolo de transporte específico. Na verdade, você pode usar SOAP via HTTP, SMTP ou FTP. Uma idéia promissora é, portanto, usar SOAP sobre BEEP.

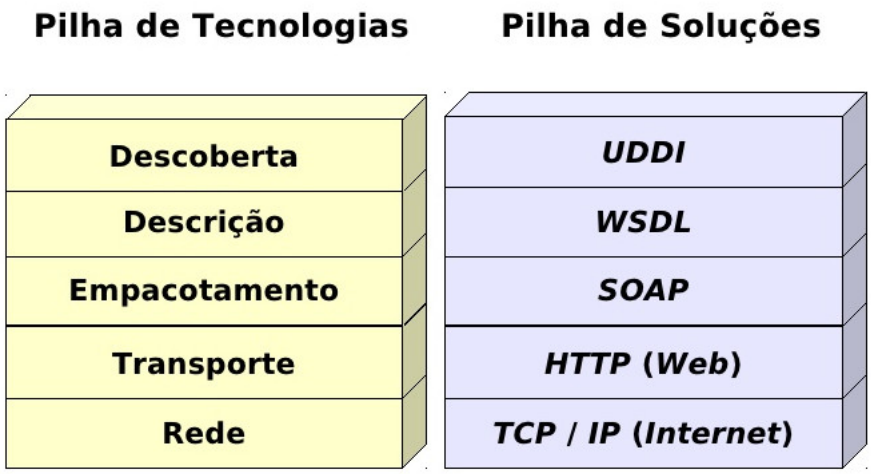

Figura 1. Pilha de protocolo de transporte e suas tecnologias.

2.3. Componentes de um *Web Service*

Ao longo dos últimos anos, três tecnologias primárias emergiram como padrões mundiais que constituem o núcleo da tecnologia de serviços da Web de hoje. Essas tecnologias são demostradas abaixo.

2.3.1. WSDL - *Web Services Description Language*

O WSDL é um idioma baseado em XML para descrever os serviços da Web e como acessá-los.

1. O WSDL foi desenvolvido conjuntamente pela Microsoft e pela IBM.
2. WSDL é um protocolo baseado em XML para troca de informações em ambientes descentralizados e distribuídos.
3. WSDL é o formato padrão para descrever um serviço web.
4. A definição WSDL descreve como acessar um serviço da Web e quais as operações que ele executará.
5. O WSDL descrever como se relacionar com serviços baseados em XML.
6. WSDL é parte integrante do UDDI, um registro de negócios mundial baseado em XML.
7. WSDL é o idioma que UDDI usa.

2.3.2. SOAP - *Simple Object Access Protocol*

O SOAP é um protocolo baseado em XML para trocar informações entre computadores.

1. O SOAP será desenvolvido como um padrão W3C.
2. O SOAP é um protocolo de comunicação.
3. O SOAP é para comunicação entre aplicativos.
4. O SOAP é um formato para enviar mensagens.

5. O SOAP é projetado para se comunicar via Internet.
6. O SOAP é independente da plataforma.
7. O SOAP é independente da linguagem.
8. O SOAP é simples e extensível.
9. O SOAP permite que você percorra os firewalls.

2.3.3. REST - *Representation State Transfer*

É uma arquitetura criada para ser mais simples de se usar que o SOAP. Este pode ser usado em vários formatos de texto, como CSV (Comma-separated Values), RSS (Really Simple Syndication), JSON e YAML. Porém, só pode ser utilizado com o protocolo HTTP/HTTPS, por exemplo utilizando os métodos GET, POST, PUT e DELETE.

1. Melhor curva de aprendizado.
2. Mensagens menores e mais eficientes como o formato JSON comparado com XML.
3. Os dados podem ser colocados em cache, retornando sempre a mesma resposta para a mesma requisição.
4. Mais rápido pois precisa de menos processamento que o SOAP.

2.4. Exemplo Ilustrativo de um Web Service

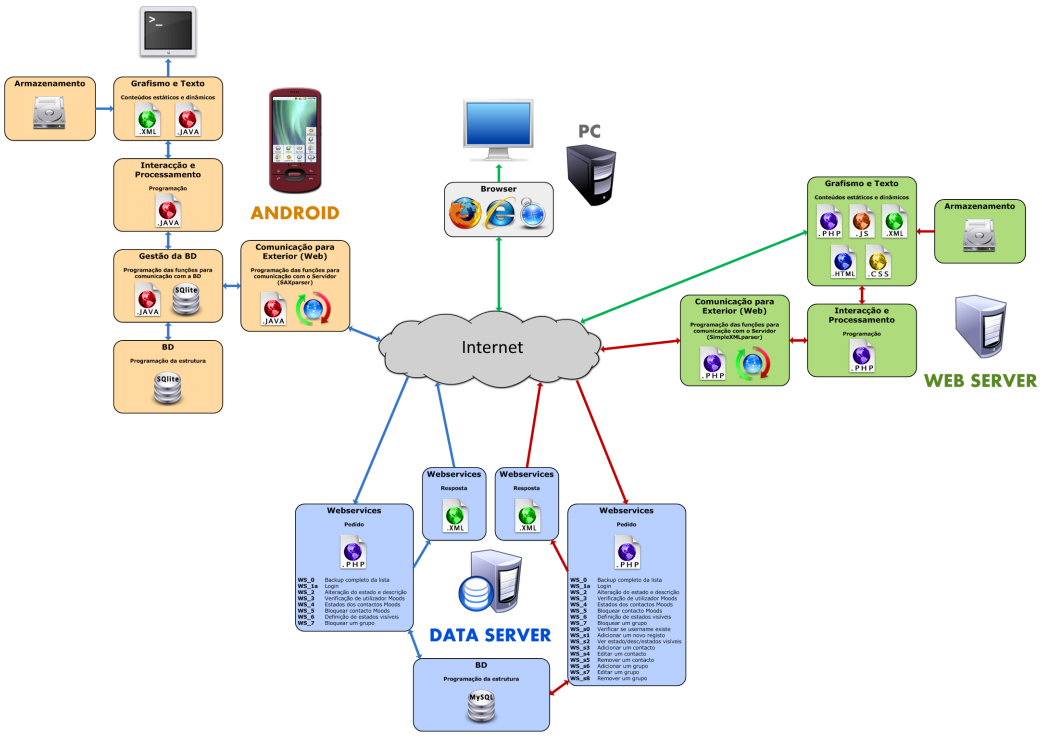

Figura 2. Sistema implantando fornecendo serviços para diversos sistemas clientes diferentes.

3. Metodologia

Nesta sessão será relatado como foi o desenvolvimento e a implantação de um *Web Service* em um *Web Server* que realiza busca de CEPs e rastreamento de encomendas dos correios.

3.1. Web Server

O *Web Server* foi implementado utilizando o NodeJS com o framework ExpressJs, onde cria um ambiente de desenvolvimento para criação de Web Services.

> O Node.js usa um modelo de E/S não bloqueante, que o torna leve e eficiente. O ecossistema de pacotes Node.js, npm , é o maior ecossistema de bibliotecas de código aberto do mundo [2].
>
> O Express é um framework para aplicativo da web do Node.js mínimo e flexível que fornece um conjunto robusto de recursos para aplicativos web e móvel [3].

3.2. Web Service de Busca de CEP

A busca de ceps esta sendo tercerizada por um *Web Service* que prove consultas de CEPs, nos mais diversos tipos de empacotamento. Esses serviços são fornecidos pela impresa *VIACEP*, o site da mesma é https://viacep.com.br/.

Uma página Web RafaelBuscas foi construida utilizando o framework Angular, a mesma consome o serviço de buscas de cep, no formato JSON fornecido pelos servidores da Viacep.

A url do Web Service da VIACEP, consumido pela página RafaelBuscas, foi o http://viacep.com.br/ws/01001000/json/, o mesmo retorna uma resposta em JSON, com as informações do CEP (Localização, Bairro, Código IBGE, GIA), para mais informações sobre requisições consultar [4].

3.3. Web Service Encomendas dos Correios

O rastreiamento por encomendas dos correios foi feita com duas tecnologias diferentes de empacontamento, fornecidos pelos Servidores dos Correios, um utiliza a tecnologia SOAP e outra alternativa e utilizar um requisição REST HTTP-POST e parsear a resposta de uma página HTML, cada uma das opções serão discutidas abaixo.

3.3.1. Serviço de Rastreamento Encomenda via SOAP

O WSDL dos correios que fornece recursos para os serviços de requisição SOAP.

O Serviço de Rastreamento de Encomendas dos correios via SOAP é gratuito.

A Url ”http://webservice.correios.com.br:80/service/rastro”é o endereço do *Web Service* e não do wsdl, o webservice espera uma mensagem SOAP como na Listing 3 a mesma deve ser assinada com cabeçalhos HTTP no formato ”Content-Type”, ”text/xml”.

```
1 "<soapenv:Envelope xmlns:soapenv=\"http://schemas.xmlsoap.org/soap/
      envelope/\" xmlns:res=\"http://resource.webservice.correios.com.
      br/\">
2                   <soapenv:Header/>
```

```
<soapenv:Body>
  <res:buscaEventos>
    <usuario>ECT</usuario>
    <senha>SRO</senha>
    <tipo>L</tipo>
    <resultado>T</resultado>
    <lingua>101</lingua>
    <objetos>RY907728402SG</objetos>
  </res:buscaEventos>
</soapenv:Body>
</soapenv:Envelope>
```

Listing 1. Código XML SOAP para requisição de localização de pacote

Utilizando a linguagem Java e a biblioteca *javax.xml.soap* foi feita uma simulação de envio de requisição SOAP com a mensagem Listing 3 para o *Web Service* de rastreamento de encomendas, fornecido pelos correios para mais informações encentivo consultar [5].

O retorno da requisição é um código XML contendo a resposta da requsição, como demostrano na Listing 4.

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Header>
    <X-OPNET-Transaction-Trace:X-OPNET-Transaction-Trace xmlns:X-OPNET-Transaction-Trace="http://opnet.com">pid=131564,requestid=6039 e4d6630ff66f79faef6849a968e0177c404c7e65d86e</X-OPNET-Transaction-Trace:X-OPNET-Transaction-Trace>
  </soapenv:Header>
  <soapenv:Body>
    <ns2:buscaEventosResponse xmlns:ns2="http://resource.webservice.correios.com.br/">
      <return>
        <versao>2.0</versao>
        <qtd>1</qtd>
        <objeto>
          <numero>RY907728402SG</numero>
          <sigla>RY</sigla>
          <nome>REM ECON TALAO/CARTAO COM AR DIGITAL</nome>
          <categoria>REMESSA ECONOICA TALAO/CARTAO</categoria>
          <evento>
            <tipo>RO</tipo>
            <status>01</status>
            <data>02/10/2017</data>
            <hora>13:49</hora>
            <descricao>Objeto encaminhado </descricao>
            <local>CENTRO INTERNACIONAL</local>
            <codigo>81010971</codigo>
            <cidade>CURITIBA</cidade>
            <uf>PR</uf>
            <destino>
              <local>Fiscalizacao Receita Federal Do
```

```
                                        Brasil</local>
27                                    <codigo>00002991</codigo>
28                                </destino>
29                            </evento>
30                        </objeto>
31                    </return>
32                </ns2:buscaEventosResponse>
33            </soapenv:Body>
34 </soapenv:Envelope>
```

Listing 2. Resposta da Requisição SOAP

Esse serviço disponibilizado pelos correios apesar de ser gratuito somente retorna a última atualização do objeto, como demostrado na Listing 4.

3.3.2. Serviço de Rastreamento de Encomendas alternativo utilizando Crawling na Página dos Correios

Existe um serviço disponível na página dos correios para rastreamento de encomendas utilizando esse serviço da página web é possivel fazer um Crawling da página, com o objetivo de extrair as informações da encomenda.

O serviço está disponível na url "http://www2.correios.com.br/sistemas/rastreamento/", a imagem 3 demostra como pode-se utilizando a ferramenta *Postman* para realizar uma requsição Rest HTTP-POST, direto para página de rastreamento com objetivo de extrair informações da mesma.

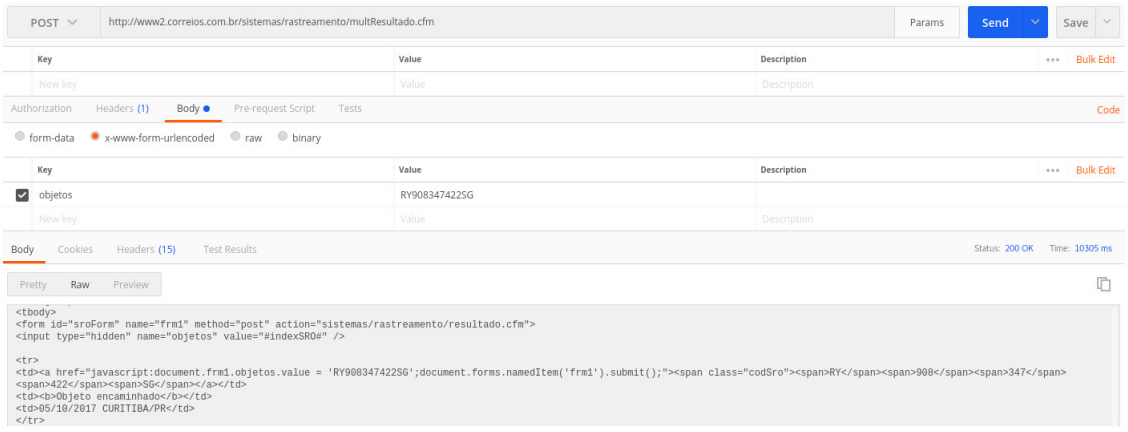

Figura 3. Postman realizando um requsição REST HTTP-POST.

Para consumir o serviço da página dos correios primeiramente temos que criar um *Web Service* com NodeJs e o framework ExpressJs, junto com a biblioteca request-promise para requisição JavaScript e a biblioteca cheerio para realizar o parse das tag do HTML.

Na Listing 3 foi criado um método que realiza um requisição REST - HTTP-POST no *Web Server* da página web.

```
request: function (req) {
var correios = {
uri: "http://www2.correios.com.br/sistemas/rastreamento/
    multResultado.cfm",
form: {
objetos: req.params.code
},
method: 'POST',
headers: {}
};
    response = requestPromise(correios);
    console.log(response)
    return response
}
```

Listing 3. Criando Requisição em Java Script

Após a requsição da Listing 3, um variável data contendo todo o HTML da requsição e parseado pela função na Listing 4, com o objetivo de extrair as informações necessárias.

O parse percorre a tabela e extrai, algumas informações (código, localidade, data, situação) com objetivo de criar um objeto JSON, com as informações de rastreamento que será consumido pela página web (RafaelBuscas).

Sendo assim uma requisição através da página RafaelBuscas ou utilizando diretamente a url http://rafaelbuscas.ddns.bet/json/"codigo-de-rastreamento", primeiramente e recebida pelo *Web Service* que está aguardando na rota da url acima, após recebido uma requisição o mesmo requisita a página de rastreamento dos correios com o código fornecido, e depois realiza um parse da página extraindo as informações e encaminhando o objeto em JSON como response para requisição inicial.

```

parser: function (data) {
    var $ = cheerio.load(data);
    var objetos = [];
    var tableObjetos = $('table').find('tr');
    $(tableObjetos).map(function(key, objeto){
        objeto  = $(objeto).children('td').map(function (key,
            field) {
            return $(field).text();
        }).toArray();
        if (objeto[0]) {

            var rastreio = {
                codigo  : null,
                situacao: null,
                local   : null,
                data    : null
            };

            rastreio.codigo     = objeto[0].trim();
            if (objeto[2]) {
                rastreio.situacao   = objeto[1];
                rastreio.local      = objeto[2].substr(11, objeto
```

```
                    [2].length).trim();
23                  rastreio.data       = objeto[2].substr(0,10).trim();
24              } else {
25                  rastreio.situacao   = 'Objeto ainda nao consta no
                        sistema'
26              }
27
28              objetos['rast'] = rastreio;
29          }
30      });
31
32      return Object.assign({}, objetos);
33  }
```

Listing 4. Criando um Crawling para parsar as tags

4. Página RafaelBuscas

Para fins didaticos os serviços criados nesse relatório estão disponíveis na url http://rafaelbuscas.ddns.net.

Essa página está hospedada em uma máquina virtual do Google Cloud, a mesma está disponível para buscas de cep e rastreamento de encomendas sem custos. Na página inical no canto inferior tem um input para colocar o código do CEP ou do Rastreamento, um componente para seleção de Cep ou Rastremamento com nome de TIPO, tem que ser marcado como CEP ou Rastreio, antes da busca(butão com uma lupa) como demostrado na Figura 4.

Figura 4. Página RafaelBuscas, demostração de busca de cep.

5. Conclusão

Esse relatório demostro como pode ser criado uma página web e um *web service* que consome serviços de outros *Web Services* disponíveis por outras impresas, ou utilizar os

mesmos para criação de um novo serviço, para atender as mais diversas necessidades.

Referências

[1] T. Spoint, "Web Services - Examples." www.tutorialspoint.com/webservices/web_services_examples.htm, 2017. [Online; acesso em 05-Outubro-2017].

[2] NodeJs, "Documentação de referência da API." https://nodejs.org/en/docs/, 2017. [Online; acesso em 02-Outubro-2017].

[3] ExpressJs, "Documentação de referência da API." http://expressjs.com/pt-br/4x/api.html, 2017. [Online; acesso em 05-Outubro-2017].

[4] VIACEP, "Códigos de Endereçamento Postal (CEP) do Brasil." https://viacep.com.br/, 2017. [Online; acesso em 05-Outubro-2017].

[5] Correios, "Web Service de Rastreamento." https://www.correios.com.br/para-voce/correios-de-a-a-z/pdf/rastreamento-de-objetos/manual_rastreamentoobjetosws.pdf, 2017. [Online; acesso em 05-Outubro-2017].