# ABORDAGEM RESTFUL EM SERVIÇOS WEB

ANDRÉ LUIZ KUCZNER PEREIRA, ANDREI RICARDO RODRIGUES, ELINEY SABINO, RAMON ALVES TRIGO, THISSIANY BAETRIZ ALMEIDA, LUIZ CLAUDIO BARRETO.

**RESUMO**

O presente artigo visa mostrar as principais características desses serviços com base na arquitetura **REST** ou **RESTful**, que tem sido adotada pelas maiorias das empresas nesses últimos anos, mostrando a sua arquitetura e quais as vantagens de utilizar esse tipo de arquitetura com foco na aprendizagem de alunos de módulo de desenvolvimento web do curso de análise e desenvolvimento de sistemas.

**Palavras Chaves:** RESTful, Tecnologias, Desenvolvimento Web

## 1. INTRODUÇÃO

Uma das questões relevantes para os estudos de tecnologias em curso superiores é o que se ensina da na grade do curso [Amem and Nunes 2006]. Isso faz com que se tenda para o mais importante que é a base da informação. Por exemplo em um módulo de desenvolvimento em linguagens web, foca-se na construção de páginas web e uma linguagem avançada como **PHP** [Niederauer 2004], mas outras questões acabam ficando de fora da grade curricular. Este trabalho traz um dos pontos que se entende ser de grande importância como leitura introdutória complementar, justamente para o módulo de desenvolvimento web, para o qual este trabalho contempla um artigo sobre a arquitetura **REST** conhecida também como **RESTful**.

## 2. METODOLOGIA

Baseando-se em documentos online, artigos e livros, usou-se estes conhecimentos para a produção deste artigo unindo a tecnologia de coleta de dados da Google, o Goole Trends, para levantar dados na forma de termos de interesse de pesquisa e busca sobre o assunto e aprendizagem na internet. Com o levante dos conhecimentos adquiridos, organizou-se um documento introdutório as arquiteturas de serviços web [Nunes and David 2005] com ênfase na arquitetura REST (Representational State Transfer, em português Transferência de Estado Representacional) [Feng et al. 2009] comentando também a arquitetura SAOP

(Simple Object Access Protocol, em português Protocolo Simples de Acesso a Objetos) [Colan 2004] e as web services. Com tudo, ainda foi apresentado um paralelo entre as duas arquiteturas (ver seção 6) mostrando a importância das arquiteturas de web services, tanto para organização de trabalho, segurança, quanto na facilidade do desenvolvimento de qualquer aplicação de serviços presentes na internet.

## 3. SERVIÇOS WEB

Existem diversas definições de serviços Web. Segundo Austin [OTTONI 2002] um serviço Web é uma aplicação autônoma, identificada por meio da URI (Uniform Resource Identifier), onde as interfaces e ligações são encontradas e descritas por meio de entidades que fazem uso da **XML** (*Extensible Markup Language*). Um serviço Web interagi com outras aplicações via da troca de dados, utilizando protocolos de comunicação padrão que estão atualmente disponíveis na Internet. Os serviços Web podem estar ligados a domínios de confiança. Desta forma, Kaye [Kaye 2003] os classifica como serviços Web internos e externos. Os primeiros estão relacionados a um único domínio de segurança, geralmente a própria empresa ou uma Intranet. Os serviços Web externos estão, normalmente, conectados a mais de um domínio ampliando as fronteiras da condução de negócios através da Internet. Em última análise, eles refletem processos de negócio dos parceiros através da infraestrutura da rede, além de possibilitar a interoperação de sistemas produzidos por fabricantes e tecnologias distintas. [da Cruz 2005]

De acordo com Kaye [Kaye 2003] os serviços *Web* apresentam inúmeros benefícios, dentre os quais destacamos: independência de plataforma de *hardware* e software; baixo acoplamento devido à elevada granulosidade dos módulos e reusabilidade dos módulos característicos que aumenta a velocidade de integração destes. Finalmente, a ubiquidade, a padronização e a escalabilidade dos serviços Web são diferenciais importantes frente a tecnologias, como, **CORBA**, **DCOM** e **RMI**. Por tanto, o modelo aplicado a *web services* mais usado no mundo é o **REST**. Para exemplificar essa preferência foi utilizado a **API** da GOOGLE, o GOOGLE TRENDS (ver figura 1) para verificar se o termo **RESTful** é realmente maior que seu principal concorrente o **SOAP** em relação aos *web services*.

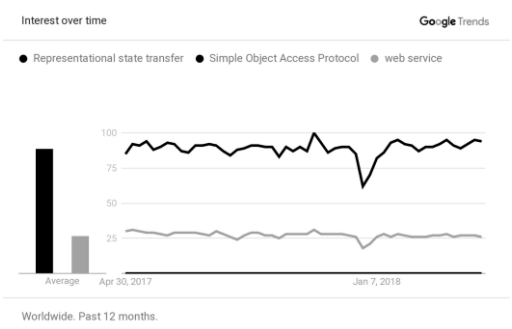

Figura 1. Resposta da timeline para REST, SOAP e web service maio de 2018. Pela ordem leia-se: REST (Linha superior), SOAP (Linha do meio) e o termo Web Service (Última linha). Fonte: [Rodrigues and Trends 2018]

## 4. SERVIÇOS WEB NO MODELO SOAP

Segundo Cruz [da Cruz 2005], os serviços Web são baseadas como arquiteturas de serviços, uma dessas arquiteturas é conhecida amplamente como **SOA** (*Service oriented Architecture*). Nesta arquitetura compreendem uma coleção de serviços que conversam através da troca de mensagens **XML**. Neste tipo de arquitetura estão definidos três papéis que trocam dados entre si. [Ferrari et al. 2007]

Segundo Cruz [da Cruz 2006] esses papeis são:

1. Provedor de serviço – responsável pela descrição e publicação de um determinado serviço Web dos serviços. O provedor também é responsável por descrever as informações de ligação do serviço usadas para sua chamada.

2. Consumidor do serviço – responsável por descobrir um serviço, obter a sua descrição e, usá-lo para se conectar a um provedor invocado um serviço Web;

3. Registro dos serviços - mantém um diretório com informações sobre serviços, como exemplo, nome, provedor e a categoria.

## 5. SERVIÇOS WEB NO MODELO REST OU RESTFUL

O modelo **REST** é bem simples, ele representa a estrutura que os dados são transportados através do protocolo **HTTP** idealizados por Roy Fielding em 2000. [Fielding and Taylor 2000] (ver figura 2) é uma abstração da arquitetura da **World Wide Web**, mais precisamente, é um estilo arquitetural que consiste de um conjunto coordenado de restrições arquiteturais aplicadas a componentes, conectores e elementos de dados dentro de um sistema de hipermídia distribuído.

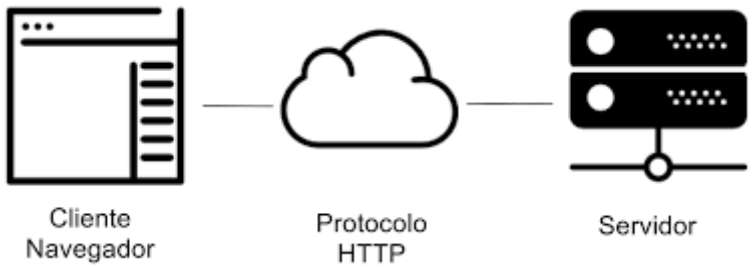

Figure 2. Arquitetura REST.Fonte [de Sousa and Almeida 2017]

A base da arquitetura **REST** está apoiada em quatro princípios: (**i)** A identificação dos recursos, os quais precisam de um nome para serem encontrados, geralmente se faz com o uso de uma ID, (**ii**) O uso de operadores de métodos padronizados pelas linguagens pertinentes e protocolos através da **url** como mostrado na tabela 1, (**iii**) uso da representação dos recursos, isso se da na forma como a **API** daquela *web service* responde a requisição do cliente, assim pode ser apresentado em formatos interfaceáveis como **xml** e **JSON** e por fim (**iv**) o principio de vinculo, onde usa-se *hypermídias* como **HATEOAS** (*Hypermedia as the Engine of Application State*). O **REST** ignora os detalhes da implementação de componente e a sintaxe de protocolo com o objetivo de focar nos papéis dos componentes, nas restrições sobre sua interação com outros componentes e na sua interpretação de elementos de dados significantes. Desta forma o cliente faz uma requisição pelo protocolo **HTTP** por meio de uma **API** que transporta os dados até o *web service*, quem é o responsável pelo tratamento dos dados e retorna somente as mensagens ao cliente.

**TABELA 1. MÉTODOS DOS OPERADORES. FONTE PRÓPRIA.**

| **Operação** | **SQL** | **HTTP** |
|---|---|---|
| Create | INSERT | Put/Post |
| Read | SELECT | Get |
| Update | UPDATE | Put/Path |
| Delete | DELETE | Delete |

Segundo Fielding [Fielding and Taylor 2000], a principal diferença do **REST** sobre o **SOAP** é a uniformidade das interfaces de comunicação entre **API**, *web service* e o cliente, onde no **SOAP** à necessidade de ser criar uma interface específica para cada aplicação não havendo a figura da **API**. Outra questão é a apresentação das camadas dos modelos no **SOAP** a interface é definida e apresentada junto aos dados e no **REST** a interface fica junto a camada de transporte deixando os dados para ser representados de forma simples e organizada.

## 6. RESULTADOS

O protocolo de serviços **SOAP** é amplamente utilizado no mundo com bastante influência sobre a lógica de negócios de muitas empresas pelo mundo todo fazendo frente ao modelo **RESTful**. Embora as duas tecnologias sejam muito bem aproveitadas, nota-se que a arquitetura **REST** é mais atraente do ponto de vista de desenvolvimento por ser uma abstração representativa e simples com elegância. Portanto até a data de levantamento pela ferramenta de buscas por *trends*, o modelo **RESTful** tem maior preferência por aqueles que trabalham com serviços na *internet*.

## 7. CONCLUSÕES

A web se renova constantemente e sua tecnologia é a impulsionadora mais relevante, pois o que a de novo é devido a facilidade que a tecnologia de fácil uso proporcionou ao desenvolvimento de empresas na internet, criando grandes aplicações de proporções inimagináveis. Mas tudo isso só foi possível com uso de modelos de comunicação e de arquitetura para esse tipo de *web services*. Portanto este trabalho trouxe a luz de um conhecimento complementar e de forma minimizada uma das arquiteturas de maior sucesso, por sua simplicidade, facilidade de aprendizado e utilização de recursos de serviços que antes eram difíceis de entender ou de padrões mais complexos, a fim de demonstrar que o **REST** é uma solução viável e que já tomou grande parte do mercado mundial para si.

## 8. REFERÊNCIAS BIBLIOGRÁFICAS

Amem, B. M. V. and Nunes, L. C. (2006). Tecnologias de informação e comunicação: contribuições para o processo interdisciplinar no ensino superior. Revista Brasileira de Educação Médica, 30(3):171–180.

Colan, M. (2004). Service-oriented architecture expands the vision of web services, part 1. IBM DeveloperWorks, April.

da Cruz, S. M. S. (2005a). Serviços web–uma breve introdução (parte i) -continuação.

da Cruz, S. M. S. (2005b). Serviços web uma breve introdução. http://www.nce.ufrj.br/conceito/artigos/2005/01p1-1.htm/. Acessado em: 13 maio de 2018.

de Sousa, A. R. R. and Almeida, W. H. C. (2017). Capítulo 15 desenvolvendo aplicações restful utilizando node.js. Livro Anais - Artigos e Minicursos 978-85-7669-395-6, III Escola Regional de Informática do Piauí. p. 532-548.

Feng, X., Shen, J., and Fan, Y. (2009). Rest: An alternative to rpc for web services architecture. In Future Information Networks, 2009. ICFIN 2009. First International Conference on, pages 7–10. IEEE.

Ferrari, R. C. C. et al. (2007). Uma solução peer-to-peer para o gerenciamento da distribuição de dados baseada na arquitetura de alto nível hla.

Fielding, R. T. and Taylor, R. N. (2000). Architectural styles and the design of networkbased software architectures, volume 7. University of California, Irvine Doctoral dissertation.

Kaye, D. (2003). Loosely coupled: the missing pieces of Web services. RDS Strategies LLC.

Niederauer, J. (2004). Desenvolvendo websites com php. Sao Paulo: Novatec.

Nunes, S. and David, G. (2005). Uma arquitetura web para serviços web. XATA-2005.

OTTONI, P. (2002). John langshaw austin and the performative view of language. DELTA: Documentação de Estudos em Linguística Teórica e Aplicada, 18(1):117– 143.

Rodrigues, A. R. and Trends, G. (2018). https://trends.google.com.br/trends/explore?q=%2Fm%2F03nsxd,%2Fm%2F077dn,Web%20service. Acessado em: 8 maio de 2018.