Proposta para trabalho de graduação

Aluno: Filipe de Alencar Ximenes

Professor orientador: Fernando José Castor de Lima Filho

Data: 30/10/2014

Cronograma

31/10/2014 - Entrega da proposta de trabalho.

14/11/2014 - Protótipo funcional fazendo requisições simples a API do Facebook.

28/11/2014 - Adaptação do protótipo do facebook para que funcione também com a API do Twitter

06/12/2014 - Suporte a hypermedia.

19/12/2014 - Suporte a paginação.

26/12/2014 - Integração com Instagram.

06/01/2015 - Exception handling.

09/01/2015 - Ultimos ajustes da ferramenta

16/01/2015 - Introdução e Contexto do trabalho escrito

06/02/2014 - Corpo do trabalho

13/01/2015 - Entrega do trabalho escrito.

Ferramenta para construção de invólucros de APIs Web

Estamos na era dos serviços na internet, a todo instante surgem novas empresas fornecendo software cada vez mais especializado. A especialização tem provocado nestes serviços algo similar à filosofia do Unix [1] onde o poder das ferramentas é gerado a partir da composição dos programas e não da construção de um único com todas as funcionalidades. Este tipo de arquitetura de software é conhecido como SOA (Service Oriented Architecture) [2]

Na internet, a forma de se “compor” programas são as APIs Web. Estas APIs utilizam em sua maioria o mesmo protocolo que utilizamos para navegar na rede: o HTTP (*Hypertext Transfer Protocol*). Assim, cada serviço tanto disponibiliza entradas, para que outros serviços se acoplem a ele, como também implementa integrações utilizando as APIs disponibilizadas por outros serviços. Esta é uma tendência tão forte que já existem serviços especializados em integrar APIs como é o caso do ifttt [3] e do zapier [4].

Compor serviços para prover um novo, simplifica o desenvolvimento e permite que o novo serviço seja mais rapidamente programado, testado e posto no mercado. Aplicações que utilizam este tipo de arquitetura são conhecidas como *mashups* [5].

Neste contexto de serviços disponibilizando suas APIs e consumindo outras, é notório o problema da falta de padronização. Cada API possui sua própria arquitetura de dados, umas seguem os padrões arquiteturais do REST [6], permitindo a navegação através de links,

enquanto outras utilizam templates de urls [7] que devem ser preenchidos para realizar as requisições. O formato dos dados retornados também varia, podendo seguir um formato já existente, como os reconhecidos pela iana [8], ou criando um novo.

A fim de reduzir o esforço necessário para interagir com as APIs, os desenvolvedores de software criam “invólucros” os quais traduzem as peculiaridades das APIs para formatos de dados mais amigáveis à linguagem de programação que estão utilizando. Porém, apesar de simplificarem as interações, cada um destes invólucros possui sua própria arquitetura de dados. Isso faz com que o programador que vai utilizar um invólucro construído por outra pessoa tenha que, além de aprender sobre a API com a qual está trabalhando, estudar sobre como o invólucro interage e armazena os dados.

Com o intuito de padronizar a interação com os invólucros das APIs, propomos a criação de uma ferramenta que permita que o programador tenha apenas que estudar a API, e não precise aprender a lidar com o invólucro a cada nova API que vai utilizar. Esta ferramenta será implementada a princípio na linguagem de programação Python [9], mas seus conceitos poderão ser reutilizados para criação de ferramentas com o mesmo propósito em outras linguagens.

A ferramenta proposta também deve permitir a simplificação da criação de invólucros e deve estar disponível como código aberto para facilitar sua adoção. Também deve permitir que terceiros implementem seus próprios invólucros para APIs diversas, resultando em uma ferramenta consistente com as demais que utilizarem a mesma tecnologia.

Para avaliar este projeto, o ponto mais importante é comprovar que a ferramenta é extensível às APIs disponíveis na internet. Para tanto, propomos a implementação de ao menos 3 clientes de APIs de serviços populares, tais como: Twitter [10], Facebook [11] e Instagram [12].

Também é importante verificar a dificuldade de se utilizar a ferramenta para construção de novos invólucros. Para que o projeto seja bem sucedido, é preciso que a ferramenta seja adotada como o padrão para implementação de invólucros, do contrário os invólucros gerados se tornarão “mais um padrão” de arquitetura para invólucros, e se existem vários padrões, não há padrão. Um bom indício de simplicidade é a quantidade de linhas de código (LOCs) necessárias para se implementar um invólucro. Por isso, propomos uma comparação de LOCs entre um invólucro feito utilizando a ferramenta e um já existente para um determinado serviço.

Por fim, é preciso analisar se os invólucros gerados com a ferramenta têm uma interface amigável e fácil de ser compreendida pelo desenvolvedor que irá utilizá-los. Para isso, pretendemos avaliar características como a documentação necessária para explicar o uso dos invólucros.

Referências

- [1] http://en.wikipedia.org/wiki/Unix_philosophy
- [2] <http://www.opengroup.org/soa/source-book/soa/soa.htm>
- [3] <https://ifttt.com/>
- [4] <https://zapier.com/>

- [5] http://www.infosys.tuwien.ac.at/staff/sd/papers/ServicesMashups_IC.pdf
- [6] http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm
- [7] <https://tools.ietf.org/html/rfc6570>
- [8] <http://www.iana.org/assignments/media-types/media-types.xhtml>
- [9] <https://www.python.org/>
- [10] <https://dev.twitter.com/overview/documentation>
- [11] <https://developers.facebook.com/docs/graph-api>
- [12] <http://instagram.com/developer/endpoints/>