SpyREST: Automated Example Based Documentation for RESTful Web APIs

S M Sohan
Department of Computer Science
University of Calgary
Calgary, Alberta T2N 1N4
Email: http://smsohan.com/home

***Abstract*—Documentation of RESTful APIs are expensive to produce and maintain. It is expensive because there is a lack of reusable tools and automated solutions for RESTful API documentation. Most RESTful APIs are documented manually and the API developers are responsible for keeping an updated documentation as the API evolves making the process both costly and error-prone. In this paper we introduce SpyREST, a reusable tool that can automatically generate RESTful API documentation. SpyREST uses a proxy to intercept example API calls and intelligently produces API documentation for RESTful Web APIs by processing the request and response data. Using SpyREST, RESTful API developers can significantly reduce the cost of producing and maintaining API documentation by replacing a large part of the manual process of documentation with an automated one.**

Keywords*—RESTful API, Web API, Documentation, Automation, Example based documentation

I. Introduction

RESTful APIs are used as a primary interconnection mechanism among modern day Web based systems. For example, the website of a restaurant can use the RESTful API from Twitter to show the latest tweets mentioning the restaurant so that prospective customers can read the experience shared by others. To allow others to use their APIs, Twitter and other RESTful API developers publish documentation describing different features of their RESTful API. The documentation of such RESTful APIs are commonly produced and maintained using a manual process.

API documentation for library APIs, such as Java Standard Edition, commonly leverage reusable tools such as JavaDoc. The documentation produced by such tools include description of objects and methods, with custom texts primarily sourced via comments. On the other hand, RESTful APIs documentation includes additional information such as HTTP headers, request parameters, request and response data in serialized formats such as JSON, XML that cannot be easily derived from looking at the objects and methods in the source code. Using comments for these additional information also requires significant manual effort because there is a lack of reusable tools to automate the documentation process. This makes the task of RESTful API documentation a costly and error-prone one. In addition to producing the API documentation, API developers also need to publish and often maintain the documentation for multiple versions as the RESTful API evolves. This requires further manual effort because it becomes a continuous process.

To produce RESTful API documentation with information about HTTP headers, URL parameters, request and response data, the manual process is driven by example API calls. Example calls are made and recorded against an API endpoint to gather information for documentation. This process can be described as a six-step process as follows: 1) craft an example call to an API endpoint with required headers, URL parameters and request body, 2) make the call, 3) capture the response headers and data, 4) strip any unwanted data from the captured information, 5) add custom descriptions to the captured data and 6) publish the API documentation. This six-step process is essentially repeated for all API endpoints that are documented.

With SpyREST, we have implemented an innovative solution to largely automate the aforementioned manual process of RESTful API documentation so that all but steps 1 and 5 are automated. Steps 1 and 5 are left to a manual process to allow for a pragmatic solution that leverages computers to automate the repeated part of the process while leaving the rest to humans. Our solution relies on a lightweight HTTP proxy server to intercept example API calls so that all the information about HTTP headers, URL parameters, request and response data can be automatically gathered. The collected data is then processed to present the documentation for RESTful resources under a hierarchy defined by API versions, resources and actions. Sensitive data such as authentication tokens and passwords are automatically filtered and are not captured. Because SpyREST uses a HTTP proxy server, it can generate documentation for all RESTful APIs irrespective of the technology used to implement the API. The resultant is an automated yet customizable, version-aware, collaboration enabled and reusable API documentation software as a service platform that can be used to generate and maintain documentation for any RESTful API.

The implication of SpyREST is two-fold. It helps lower the cost of producing and maintaining RESTful API documentation through automation and provides a shared platform for publishing the documentation of different RESTful APIs.

The remainder of this paper is organized as follows: in the next section we provide background information on the topic of RESTful API and API documentation. Then, we discuss the related work in the following section. The implementation details of SpyREST is provided next. Then, we discuss the implications and limitations of our work in the discussion section. We present our core contributions and future work in the conclusion.

II. Related Work

Several papers have been published on areas related to RESTful API documentation that are discussed under the following two categories: RESTful API documentation and general API learnability.

A. RESTful API Documentation

To address the need for a standard format to describe RESTful APIs several related works proposed candidate specifications. For example, Danielsen et al. presented at vocabulary for documenting RESTful Web APIs called Web Interface Language (WIfL) [?]. The vocabulary comprises of these following objects, Resource, Request, Response, Representation and Parameters that can describe the structure and example usage of RESTful Web APIs. Generation of WIfL specific objects for RESTful APIs requires manual effort or bespoke implementation since a reusable automated approach is not presented. With SpyREST, we focus on automation so that reusable tools can be used to produce RESTful API documentation.

Verborgh et at. presented RESTdesc, a Resource oriented and Hyper-link based specification for describing RESTful APIs [?]. RESTdesc relies on pre and postconditions to describe the outcome of an API call so that a general purpose API client can parse RESTdesc formatted documentation to invoke API calls to specific RESTful APIs. Mangler et al. presented RDDL, a XML based specification for describing RESTful APIs [?]. RDDL descriptions are composed of resources, operations, parameters and headers. Manual effort is needed to generate RESTdesc and RDDL formatted documentations.

Kopecky et al. presented hRESTS, a machine readable micro-format to describe RESTful APIs that uses an alternate representation compared to WIfL [?]. hRESTS specification comprises of Service, Operation, Address, Method, Input, Output and Label objects to describe RESTful APIs. Automated XML transformation is used to convert a formatted HTML documentation of a RESTful API into hRESTS so that the machine readable format can be used by RESTful API clients for automated invocation. SpyREST aims to minimize the cost of a manual or custom process to generate and maintain formatted HTML documentation by leveraging an automated approach.

Maleshkova et al. presented OmniVoke, a RESTful API based invocation engine that provides an abstraction layer for RESTful API calls to multiple APIs that follow different conventions [?]. A general purpose RESTful API client can be used against OmniVoke since the different conventions are wrapped under a uniform RESTful API.

Myers et al. performed a user study based on the documentation of Web APIs used in large scale enterprises to understand the desirable contents for Web API documentation [?]. They recommended providing a consistent look-and-feel with explanation for the starting points and an overall map comprising of both text and diagrams, providing a browsing experience with breadcrumb trail following a hierarchy, an effective search interface, providing example code and a way to exercise the examples online without writing code. SpyREST aims to achieve these desired contents and organization for RESTful APIs using an automated and customizable approach that can be reused across different RESTful APIs.

In addition to the research community, there are several RESTful API specification formats that are observed in the industry. Swagger is a JSON based specification that allows users to describe RESTful APIs in terms of paths, parameters, request and response headers and bodies [1]. RAML is a RESTful API documentation format that uses YAML files to describe RESTful API documentation using a similar hierarchy as Swagger [2]. Blueprint is another RESTful API specification format that uses Markdown files with additional tags to describe RESTful API objects [3]. In addition to producing RESTful API documentation, there are tools and software as a service (SaaS) providers that can be used to publish the documentation and auto generate API client code for RESTful APIs that are described using one of these formats. The key limitation of these tools is a manual process that is required to create and maintain the documentations for these formats since there is a lack of automated approaches to do so.

III. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.

[1] https://github.com/swagger-api/swagger-spec/blob/master/versions/2.0.md
[2] http://raml.org/spec.html
[3] https://github.com/apiaryio/api-blueprint