International Journal of Current Engineering and Technology
©2021 INPRESSCO®, All Rights Reserved

E-ISSN 2277 – 4106, P-ISSN 2347 – 5161
Available at http://inpressco.com/category/ijcet

Research Article

SAS Command Line Interface Code Generator

Ritupriya Andurkar[1], Pravin S. Game[2], Rakesh Jadhav[3] and Arvind Jagtap[4]

[1,2]Department of Computer Engineering, Pune Institute of Computer Technology, Pune, India
[3,4]Department of Business Intelligence & Platforms R&D SAS R&D, India Pvt. Ltd Pune, India.

Received 10 Nov 2020, Accepted 10 Dec 2020, Available online 01 Feb 2021, **Special Issue-8 (Feb 2021)**

Abstract

Command line interfaces (CLIs) provide easy to use abstraction on top of REST (Representational State Transfer) APIs (Application Programming Interfaces) and they are helpful in automation. Hence CLIs are used as interfaces for administrative work. In the latest SAS platform, a CLI framework is introduced which supports a structure of commands, subcommands, and flags. Currently, CLIs are developed manually by programmers. This requires a lot of time and cost. To ease the CLI development automation is being considered as per '12-factor app' methodology which says, 'use declarative formats setup automation, to minimize time and cost for new developers joining the project'. This work involves developing a tool to generate CLI code. CLIs are mainly driven by REST endpoints. The Input will be swagger file which contains API specification and CLI code will be generated as output. One CLI command will be generated for each REST endpoint. This tool can be helpful to CLI developers for quick start with generation of boilerplate code. As most of the code will be generated automatically, efforts required for CLI development will be reduced, thus saving the involved time and cost. This tool will also help to maintain the consistency in CLI code for different services.

Keywords: *Code Generators, Command line Interfaces, Representational State Transfer (REST)*

Introduction

A code generator is a tool which generates code based on some specification. Several code generators are found in literature. They generate source codes from specifications such as UML diagrams, XML documents, etc. Command line Interface is a textbased interface between user and computer where a line of text (called command line) is passed between the two for communication [1]. An application programming interface (API) is a code that allows two software programs to communicate with each other. An API defines the correct way for a developer to request services from an operating system (OS) or other application and expose data within different contexts and across multiple channels [2]. In the latest SAS platform, a CLI framework is introduced which supports a structure of commands, subcommands, and flags. SAS APIs are based on REST architecture. REST (REpresentational State Transfer) is a distributed programming architectural style in which the World Wide Web is viewed generally as a large distributed application. In this style, the application is distributed as structured data rather than by explicitly invoking remote procedure calls on remote objects. Each data element is a resource, addressed by a URI (Uniform Resource Identifier), which may be provided in multiple representations. The state is defined by resources consumed by the client rather than being hidden behind a session identifier in a service tier. A limited vocabulary of standard verbs is used to manipulate that state. (By contrast, Web Services, whether SOAP or XML-RPC, tend to use domain-specific verbs: uniquely-named remote procedure calls or remote object methods.) A limited vocabulary of return values describes the context of the results. [3]
OpenAPI Specification (formerly Swagger
Specification) is an API description format for REST APIs. An OpenAPI file allows you to describe your entire API, including:

- Available endpoints (/users) and operations on each endpoint (GET /users, POST /users)
- Operation parameters Input and output for each operation
- Authentication methods
- Contact information, license, terms of use and other information.

API specifications can be written in YAML or JSON. [4]
Manual development of Command line interfaces (CLIs) requires a lot of time and effort. It also requires some initial learning for understanding the Application Programming Interfaces (APIs) for which CLIs are to be developed and the SAS CLI framework which is used in development of CLIs. Automation can help to ease CLI

development. Study of existing CLI code shows that there exists a pattern in CLI code for different services. They all provide functionality for common operations such as create, read, update and delete. This pattern can be used to create a template which will help in the generation of the CLI code. This will help to reduce developer‟s efforts required for CLI development and will also help to maintain consistency in CLIs of different services. In this work, a code generator will be developed which will generate command line interface (CLI) code for web services which are exposed through REST APIs. The input to the tool will OpenAPI specification (swagger document) and the output will be CLI code in Go language.

Literature Survey

Kundu et al. [5] proposed an approach to generate code from UML sequence diagrams. It consists of two steps: construction of „graph model‟ from UML sequence diagrams and code generation from the „graph model‟. It is used to generate code within class methods. The proposed approach is most effective for controller classes as compared to entity and boundary classes.

J. Jagadeesan and G.M. Kadhar Nawaz [6] proposed a mechanism to make a generalized system of MultiLanguage Source Code Generation. It takes database structure as input and produces source code in Java,
Java script, JSP, PHP and ASP. Database connectivity code and SQL queries are generated using template-based approach where application specific code is inserted in standard templates. The generated code can be utilized for both web based and pc-based applications.

S. Viswanathan and P. Samuel [7] proposed an algorithm to generate code from the combined model of activity and sequence diagrams consisting of concurrent activities. The proposed algorithm can generate class definition, method definition and control flow. A case study is also presented that demonstrates the generation of Java code for ATM operation.

J. Hoyos and F. Restrepo-Calle [8] proposed an approach to automatically generate a web application prototype running business processes using a restricted natural language specification. The code generation process consists of three phases: lexical syntactic analysis, information extraction and source code generation. This helps in fast prototyping based on initial user requirements which can further be refined by examining the generated prototype.

Sunitha E.V. and P. Samuel [9] proposed an architecture of the code generator and the process of code generation from UML state Chart diagrams. They proposed a two-phase approach: In the first phase State chart diagrams are converted to XML documents and in the second phase code is generated using this XML document. A design pattern for implementation of state diagram which includes hierarchical, concurrent, and history states is also proposed. Code generator generated code from UML state chart diagrams with the help of this design pattern.

Proposed Methodology

The development of proposed tool is based on template-based approach. In this work, templates are used which provide the skeleton for CLI code and specific tokens in these templates are replaced with values extracted from swagger specification. This leads to generation of service specific CLI code from generic templates.

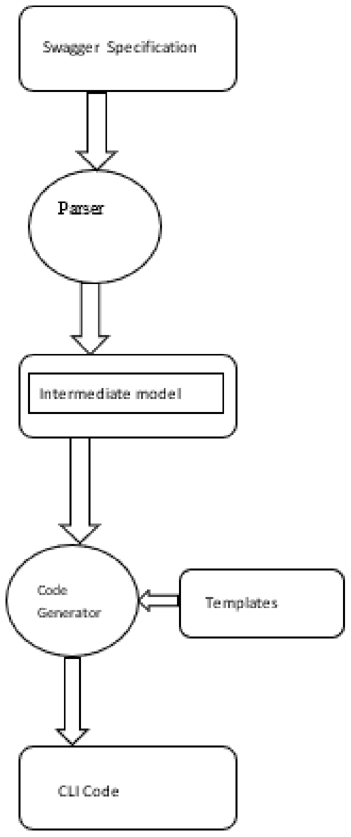

Fig 1. Block Diagram of proposed Code Generator

The input to the code generator will be OpenAPI specification (swagger document). This document will be converted to an intermediate model and then CLI code will be generated using this intermediate model and the code templates. The above figure illustrates these steps.

Conclusion

The proposed tool will generate command line interface code in go language provided an OpenAPI specification (swagger specification) of a web service. This will help to reduce the time and effort required for command line interface development. It will also help to maintain consistency in command line interfaces of different web services.

References

[1]. "Command Line Interface," available at https://www.defit.org/command-line-interface/, accessed on 15/10/2019

[2]. "Application Program Interface," available at https:// searchapparchitecture.techtarget.com/definition/ applica tion-program-interface-API, accessed on 15/10/2019

[3]. "REST," available at http://sww.sas.com/saspedia /REST, accessed on 15/10/2019

[4]. "What is OpenAPI," available at https://swagger.io /docs/specification/about/, accessed on 15/10/2019.

[5]. D. Kundu, D. Samanta, R. Mall, "Automatic code generation from unified modelling language sequence diagrams," IET Software, vol.7, Issue. 1, pp.12-28, 2013.

[6]. J. Jagadeesan, G.M. Nawaz, "Multilanguage source code Generator for Database oriented application using JSP," International Journal of Research in Engineering and Technology, Vol. 4, Issue 7, pp. 407 – 410, 2015.

[7]. S. Viswanathan, P. Samuel, "Automatic code generation using unified modelling language activity and sequence models," IET software, Vol. 10, Issue 6, pp. 164-172, 2016.

[8]. J. Hoyos and F. Restrepo-Calle, "Automatic source code generation for web-based process-oriented information systems," 12th international Conference on Evaluation of Novel Approaches to software Engineering, pp. 103 – 113, 2017.

[9]. Sunitha E.V. and P. Samuel, "Automatic Code Generation from UML State Chart Diagrams," IEEE Access, Vol. 7, pp. 8591- 8608, 2019.