**OpenAPI Generator**



Software Systems Architecture and Integration

January 2021

Víctor Graván Bru (vgravanbru@gmail.com)

Juan Antonio Ortiz Guerra (juanantonioortizguerra@gmail.com)

María del Mar Vallejo Gamboa (vallejo.mmar@gmail.com)

Alberto Antonio Tokos Matas (belatm12@protonmail.com)

Nuria Gómez Arias (nuriafyq23@gmail.com)

Tutor: Sergio Segura Rueda

Group number: L6 - 02

Link to the project in GitHub: <https://github.com/OpenAPITools/openapi-generator.git>

VERSION HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Description | Participants |
| 07/01/2021 | 1.0 | - Initial version. |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Índice

[1 Introducción 4](#_Toc63239815)

[2 Visión general 4](#_Toc63239816)

[3 Participantes 4](#_Toc63239817)

[4 Vistas 4](#_Toc63239818)

[4.1 Vista de contexto 4](#_Toc63239819)

[4.2 Escenarios de uso 4](#_Toc63239820)

[4.3 Vista functional 4](#_Toc63239821)

[4.4 Vista de despliegue 4](#_Toc63239822)

[4.5 Vista de Desarrollo 4](#_Toc63239823)

[5 Puntos de variabilidad y extensión 5](#_Toc63239824)

[6 Análisis de atributos de calidad 5](#_Toc63239825)

[7 Sugerencias de mejora 5](#_Toc63239826)

[8 Contribuciones al proyecto 5](#_Toc63239827)

[9 Conclusiones 5](#_Toc63239828)

[Referencias 5](#_Toc63239829)

# Introducción



**OpenApi** originalmente conocido como especificación [Swagger](https://es.wikipedia.org/wiki/Swagger_(software)), es una especificación que proporciona interfaces entendibles por máquina para realizar, consumir, describir y visualizar servicios web [RESTful](https://es.wikipedia.org/wiki/Transferencia_de_Estado_Representacional).

Comenzó sinedo parte del marco Swagger pero más adelante se transformo en un proyecto separado a partir del 2016, supervisados por la iniciativa OpenApi, el cual colabora con código abierto en la fundación Linux swagger y otras herramientas para generar código, documentación y casos de prueba.

|  |  |  |
| --- | --- | --- |
| **Versión** | **Fecha** | **Notas** |
| 3.0.3 | 2020-02-20 | Lanzamiento de parche de la especificación OpenAPI 3.0.3 |
| 3.0.2 | 2018-10-08 | Lanzamiento de parche de la especificación OpenAPI 3.0.2 |
| 3.0.1 | 2017-12-06 | Lanzamiento de parche de la especificación OpenAPI 3.0.1 |
| 3.0.0 | 2017-07-26 | Lanzamiento de la especificación OpenAPI 3.0.0 |
| 2.0 | 2014-09-08 | Lanzamiento de Swagger 2.0 |
| 1.2 | 2014-03-14 | Publicación inicial del documento formal |
| 1.1 | 2012-08-22 | Lanzamiento de Swagger 1.1 |
| 1.0 | 2011-08-10 | Primera versión de la especificación Swagger |

**OpenApi generator**

En julio del 2018, Willian Chen, el mayor contribuyente de Swagger Codegen y más de 40 colaboradores de Swagger decidierón bifurcar todo el código hacia un proyecto llamado OpenApi Generator bajo la organización OpenApi Tools.

**La principal diferencia entre OpenApi y OpenApi Generator:**

1. OpenApi: se centra en las especificaciones.
2. OpenApi generator: genera todo el código a partir de OpenApi mediante una serie de instrucciones que le puede dar el usuario.

# Visión general

OpenAPI Generator proporciona generación de librerías para clientes API (generación de SDK), server stubs (implementaciones API de testeo), documentación y configuración de forma automática dada una OpenAPI Spec (están soportadas tanto 2.0 como 3.0). Actualmente, están soportados los siguientes lenguajes/frameworks:

|  |  |
| --- | --- |
|  | **Lenguajes/Frameworks** |
| **Clientes API** | ActionScript, Ada, Apex, Bash, C, C# (.net 2.0, 3.5 or later, .NET Standard 1.3 - 2.0, .NET Core 2.0, .NET 5.0. Libraries: RestSharp, HttpClient), C++ (cpp-restsdk, Qt5, Tizen, Unreal Engine 4), Clojure, Crystal, Dart, Elixir, Elm, Eiffel, Erlang, Go, Groovy, Haskell (http-client, Servant), Java (Jersey1.x, Jersey2.x, OkHttp, Retrofit1.x, Retrofit2.x, Feign, RestTemplate, RESTEasy, Vertx, Google API Client Library for Java, Rest-assured, Spring 5 Web Client, MicroProfile Rest Client), k6, Kotlin, Lua, Nim, Node.js/JavaScript (ES5, ES6, AngularJS with Google Closure Compiler annotations, Flow types, Apollo GraphQL DataStore), Objective-C, OCaml, Perl, PHP, PowerShell, Python, R, Ruby, Rust (hyper, reqwest, rust-server), Scala (akka, http4s, scalaz, sttp, swagger-async-httpclient), Swift (2.x, 3.x, 4.x, 5.x), Typescript (AngularJS, Angular (2.x - 11.x), Aurelia, Axios, Fetch, Inversify, jQuery, Nestjs, Node, redux-query, Rxjs) |
| **Server stubs** | Ada, C# (ASP.NET Core, NancyFx), C++ (Pistache, Restbed, Qt5 QHTTPEngine), Erlang, F# (Giraffe), Go (net/http, Gin), Haskell (Servant), Java (MSF4J, Spring, Undertow, JAX-RS: CDI, CXF, Inflector, Jersey, RestEasy, Play Framework, [PKMST](https://github.com/ProKarma-Inc/pkmst-getting-started-examples), [Vert.x](https://vertx.io/)), Kotlin (Spring Boot, Ktor, Vertx), PHP (Laravel, Lumen, Slim, Silex, [Symfony](https://symfony.com/), [Zend Expressive](https://github.com/zendframework/zend-expressive)), Python (Flask), NodeJS, Ruby (Sinatra, Rails5), Rust (rust-server), Scala (Akka, [Finch](https://github.com/finagle/finch), [Lagom](https://github.com/lagom/lagom), [Play](https://www.playframework.com/), Scalatra) |
| **Generadores de documentación API** | HTML, Confluence Wiki, Asciidoc, Markdown, PlantUML |
| **Ficheros de configuración** | [Apache2](https://httpd.apache.org/) |
| **Otros** | GraphQL, JMeter, Ktorm, MySQL Schema, Protocol Buffer |

# Participantes

**Adquisidores/Proveedores**

Los adquisidores son los participantes que financian el proyecto. En el caso de OpenAPI Generator son los patrocinadores, que también proporcionan la infraestructura necesaria. Algunos ejemplos son [GoDaddy](https://www.godaddy.com/?utm_source=openapi_generator&utm_medium=github_webpage&utm_campaign=sponsor) (proporciona dominios web), o [Linode](https://www.linode.com/?utm_source=openapi_generator&utm_medium=github_webpage&utm_campaign=sponsor) (proporciona una VPS (Virtual Private Server, o servidor virtual privado))

**Asesores**

No hay un rol específico de asesor en este proyecto. OpenAPI Generator posee licencia Apache License 2.0. A menos que sea requerido por ley o se acuerde por escrito, todo software distribuido bajo esta licencia se distribuye bajo el criterio “tal cual”, sin garantías o condiciones de ningún tipo, ya sea de forma explícita o implícita.

**Comunicadores/Soporte Técnico/Responsables de Mantenimiento**

Está formado por el *Comité Técnico de OpenAPI Generator* (CTOG), formado por una parte de los colaboradores y por usuarios, que se dedican a guiar a otros usuarios, realizan mejoras al generador, y revisar problemas del programa, entre otras cosas.

**Desarrolladores**

Son miembros de la comunidad (principalmente miembros del CTOG), que se dedican a mejorar el generador o crear plantillas para el generador

**Usuarios/Testeadores**

Son los que ponen a prueba y utilizan la infraestructura para generar librerías API, software de relleno para servidores o configuración de forma automática, como grandes empresas tecnológicas o redes sociales. Algunos ejemplos son [Allianz](https://www.allianz.com/), [Kubernetes](https://kubernetes.io/), o [Twitter](https://twitter.com/) Se puede encontrar una lista con todas las compañias o proyectos que lo utilizan [aquí](https://github.com/OpenAPITools/openapi-generator#4---companiesprojects-using-openapi-generator).

**Competencia**

Está formada por aquellos sistemas que ofrecen un servicio similar a OpenAPI Generator, como [generator-rest](https://www.npmjs.com/package/generator-rest)

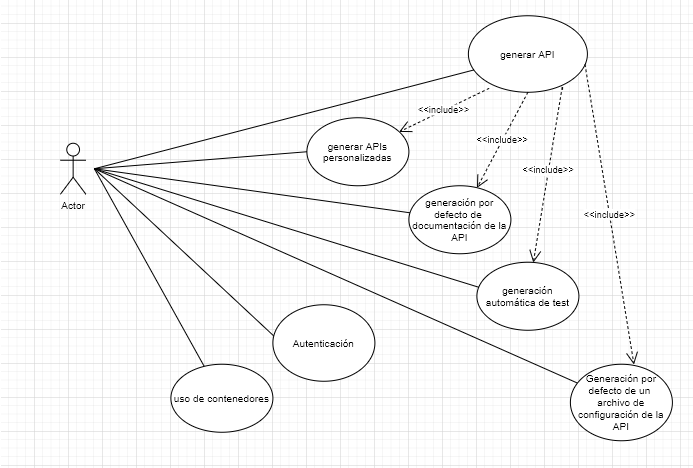
# Vistas

## Vista de contexto

La vista de contexto contiene las relaciones entre el proyecto y los participantes y dependencias involucradas en el entorno de desarrollo del proyecto. La Figura(X) muestra el diagrama de contexto de OpenApi-generator. El proyecto se encuentra en un repositorio de github, con el objetivo de facilitar el control de versiones y la documentación del proyecto, así como a su vez, llevar el manejo de las distintas issues que se puedan crear. OpenApi se trata de un proyecto de código abierto, por lo cual cualquiera puede formar parte del equipo de desarrollo. No obstante, cabe mencionar que hay un equipo principal de desarrolladores. Varias organizaciones patrocinan y usan la herramienta para automatizar la generación de APIs. El proyecto está principalmente desarrollado en Java y usa Maven para automatizar el manejo de dependencias. No obstante, se pueden apreciar otros lenguajes como JavaScript o TypeScript, haciendo posible la instalación de la herramienta de diferentes maneras. Esta multitud de posibilidades de instalación, hacen posible que la herramienta se pueda usar en cualquier sistema operativo, ya que soporta docker y vagrant, así como otras formas a través de npm y Homebrew. A su vez, la herramienta necesita de diferentes dependencias para ser testeada, tales como se pueden ver en la figura(X); y utiliza swagger como herramientas de apoyo. Finalmente, OpenApi generator, compite con otras herramientas para la autogeneración de APIs del tipo REST. Estas son rest-hapi y Generator-rest, entre otras, aunque cabe destacar que OpenApi generator mantiene cierta competencia con AsyncApi, siendo esta última usada para autogenerar APIs asíncronas, y posee una licencia de Apache.

(insertar imagen del diagrama de contexto)

## Escenarios de uso



## Vista funcional

## Vista de despliegue

La vista de despliegue describe el entorno hardware y software en el que el sistema será desplegado. En este caso, el despliegue se realiza mediante Maven, por lo que solo requiere una instalación de Java JDK (al menos Java 8) y Apache Maven (mínimo 3.3.4)

(insertar imagen de diagrama de despliegue)

## Vista de Desarrollo

# Puntos de variabilidad y extensión

# Análisis de atributos de calidad

# Sugerencias de mejora

# Contribuciones al proyecto

# Conclusiones

# Referencias

Se recomienda el uso de la herramienta [Mendeley](https://www.mendeley.com/) con sus extensiones para [navegador](https://chrome.google.com/webstore/detail/mendeley-web-importer/dagcmkpagjlhakfdhnbomgmjdpkdklff?hl=en) y [Microsoft Word](https://www.mendeley.com/reference-management/mendeley-cite).