Interactively exploring API changes and versioning consistency

SOUHAILA SERBOUT, DIANA CAROLINA MUÑOZ HURTADO, CESARE PAUTASSO

Our paper is based on APIcture, a novel tool that addresses the need for comprehensive and intuitive visualization of web API evolution. We invite the Artifact Evaluation Committee to evaluate the practicality and effectiveness of APIcture for the reproducibility of the visualization we included in our paper and published in the online gallery

1 OVERVIEW ON APICTURE

APIcture is a comprehensive tool designed to empower researchers, developers, and users to gain deeper insights into the evolution of web APIs. Our submission, APIcture, introduces an innovative approach to visualizing API changes and versioning strategies, enabling users to comprehend the temporal sequence of changes, assess compatibility issues, and understand versioning practices. The tool encompasses two main visualizations: API Changes and API Version Clock. The former provides a detailed view of changes occurring at specific time intervals, while the latter offers an overarching view of version upgrades and change patterns. Our Artifact Evaluation submission presents APIcture, detailing its functionality, utilization, and practicality to generate API evolution visualizations. We provide clear instructions on how to reproduce our visualization and explore other API evolution cases.

2 USAGE GUIDE

2.1 Installation

To start using APIcture, you need to install the CLI tool. Follow these steps:

(1) Install Node.js and npm: APIcture requires Node.js and npm (Node Package Manager) to be installed on your system. If you haven't already installed them, you can download them from the official Node.js website: https://nodejs.org/
Minimum required compatible version is: `v16.18.0`

(2) Install APIcture: Once you have Node.js and npm installed, open a terminal window and run the following command to install APIcture globally on your system:

```
npm install apict -g
```

The NPM package page is in:
https://www.npmjs.com/package/apict
To be sure that APIcture is properly installed, run: `apict –version` or `apict -v`

2.2 Basic Usage

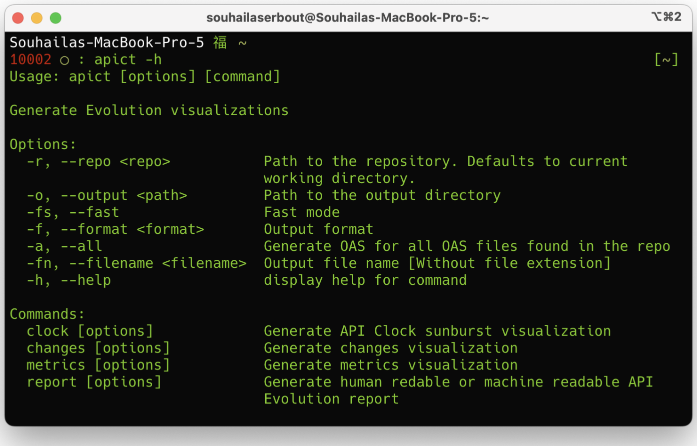
APIcture offers various subcommands that cater to different aspects of API visualization. Here are the fundamental steps to get started with APIcture:

(1) **Main subcommands**: To generate the visualizations, run the `apict` command followed by the desired subcommand and any additional options to generate the desired visualizations. The available subcommands include:
- `apict <spec-path>`: Generates visualizations for the OpenAPI specification located at the specified path.
- `apict changes <spec-path>`: Focuses specifically on changes localization.

- `apict versioning <spec-path>`: Analyzes version upgrades versus changes types.
- `apict metrics`: Generates visualizations for API metrics.

(2) **Subcommands available options**: APIcture provides several options (Figure 1) to customize the visualization process according to the users needs:

- **-r, –repo <repo>**: Specifies the path to the repository containing the API's version history. By default, the tool uses the current working directory as the repository location.
- **-o, –output <path>**: Defines the path to the output directory where the generated visualizations will be saved. If not specified, the output will be saved in the default directory.
- **-fs, –fast**: Activates the fast mode, which optimizes the execution for faster generation of visualizations. This mode is only activable if the visualization has been already generated in normal mode. If not, this option is ignored.
- **-f, –format <format>**: Specifies the desired output format for the generated visualizations. The available formats include options such as PNG, SVG, and interactive HTML.
- **-a, –all**: Generates OpenAPI specifications for all OpenAPI files found in the repository. This option facilitates generating visualizations for multiple specifications within the same repository.
- **-fn, –filename <filename>**: Specifies the output file name for the generated visualization. The tool saves the visualization with the specified file name (without file extension) in the output directory.
- **-h, –help**: Displays the help information for the command, providing a concise overview of the available options.



```
Souhailas-MacBook-Pro-5 福 ~
10002 ○ : apict -h                                                          [~]
Usage: apict [options] [command]

Generate Evolution visualizations

Options:
  -r, --repo <repo>             Path to the repository. Defaults to current
                                working directory.
  -o, --output <path>           Path to the output directory
  -fs, --fast                   Fast mode
  -f, --format <format>         Output format
  -a, --all                     Generate OAS for all OAS files found in the repo
  -fn, --filename <filename>    Output file name [Without file extension]
  -h, --help                    display help for command

Commands:
  clock [options]               Generate API Clock sunburst visualization
  changes [options]             Generate changes visualization
  metrics [options]             Generate metrics visualization
  report [options]              Generate human redable or machine readable API
                                Evolution report
```

Fig. 1. Help Command Output for APIcture Tool

2.3 Reproducibility Guide

This section provides comprehensive instructions for generating visualizations from a real world repository using the commands and options listed earlier.

For illustrative purposes, we have included sample API GitHub repositories containing OpenAPI specifications in our GitHub Repository[1].

[1]https://github.com/souhailaS/APIcture/blob/main/git_urls.json

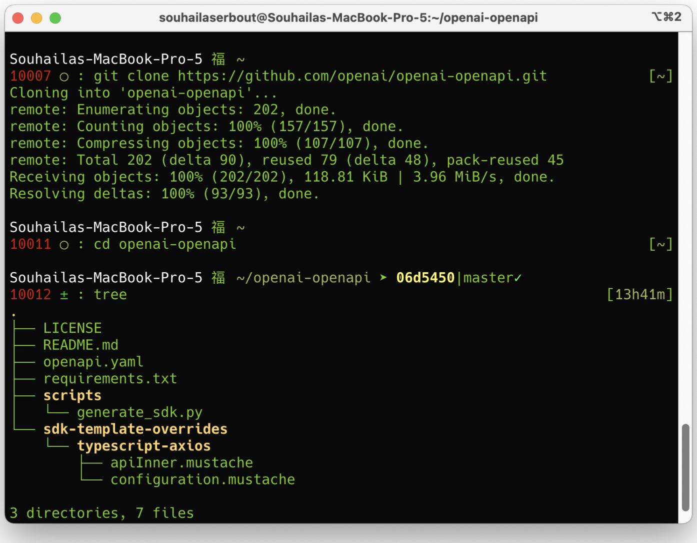
We pick the OpenAI API[2] as our example. Begin by cloning the repository to your local machine:

```
git clone https://github.com/openai/openai-openapi
```

Next, navigate into the repository:

```
cd openai-openapi
```

Figure 2 depicts the structure of the OpenAI API repository, revealing the top-level placement of the OpenAPI specification file (`openapi.yaml`).



```
souhailaserbout@Souhailas-MacBook-Pro-5:~/openai-openapi                    ⌥⌘2

Souhailas-MacBook-Pro-5 福 ~
10007 ○ : git clone https://github.com/openai/openai-openapi.git             [~]
Cloning into 'openai-openapi'...
remote: Enumerating objects: 202, done.
remote: Counting objects: 100% (157/157), done.
remote: Compressing objects: 100% (107/107), done.
remote: Total 202 (delta 90), reused 79 (delta 48), pack-reused 45
Receiving objects: 100% (202/202), 118.81 KiB | 3.96 MiB/s, done.
Resolving deltas: 100% (93/93), done.

Souhailas-MacBook-Pro-5 福 ~
10011 ○ : cd openai-openapi                                                  [~]

Souhailas-MacBook-Pro-5 福 ~/openai-openapi ➤ 06d5450|master✓
10012 ± : tree                                                           [13h41m]
.
├── LICENSE
├── README.md
├── openapi.yaml
├── requirements.txt
├── scripts
│   └── generate_sdk.py
└── sdk-template-overrides
    └── typescript-axios
        ├── apiInner.mustache
        └── configuration.mustache

3 directories, 7 files
```

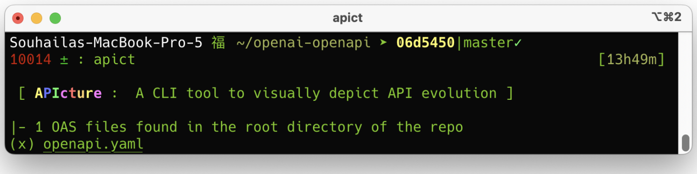
Fig. 2. Repository Structure: OpenAI API

To generate all evolution visualizations at once, simply execute: `apict`.

Alternatively, utilize the `apict` command with the `-r` option, which will run the visualizations generation without need to navigate to the repository:

```
apict -r openai-openapi
```

In the absence of a specific file path, APIcture automatically locates all OpenAPI specification files within the repository and prompts the user to select one (Figure 3).



```
apict                                                                        ⌥⌘2
Souhailas-MacBook-Pro-5 福 ~/openai-openapi ➤ 06d5450|master✓
10014 ± : apict                                                          [13h49m]

 [ APIcture :  A CLI tool to visually depict API evolution ]

|- 1 OAS files found in the root directory of the repo
(x) openapi.yaml
```

Fig. 3. `apict` Command Line: Generating Visualizations

To generate visualizations for all OpenAPI specifications within the repository, apply the `-a [-all]` option:

```
apict -r openai-openapi -a
```

[2]https://github.com/openai/openai-openapi.git

Generated visualizations are stored within the `APIcture` folder, organized under a directory named after the respective specification. For instance, in this case, the visualizations are located within `APIcture/openapi` (Figure 6). By default, if no format is given, the output format is HTML.

The generated HTML files are:

- `changes-<openapi api file name>.html`: an interactive format of the API Changes visualization.
- `version-clock-<openapi api file name>.html`: an interactive format of the API Version Clock visualization.
- `version-clock-<openapi api file name>.html`: an interactive format of six API evolution metrics plots (Figure 4).

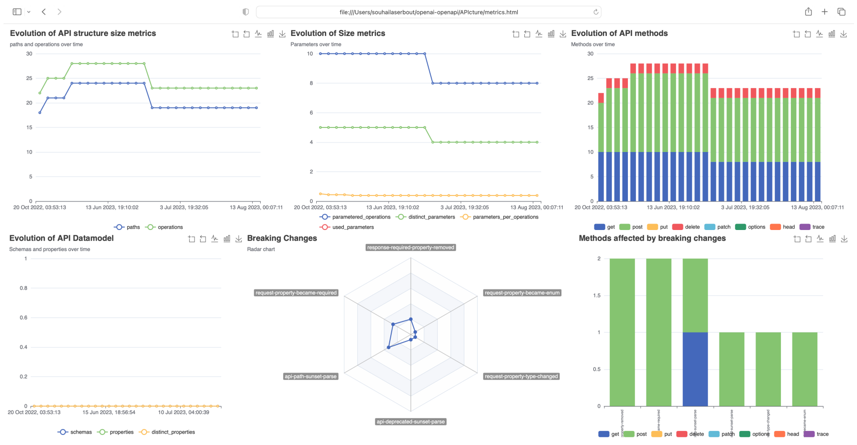
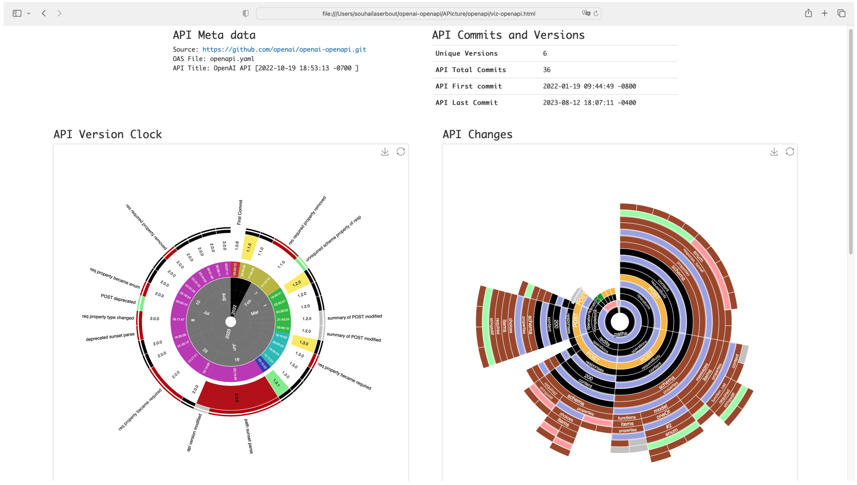

file:///Users/souhailaserbout/openai-openapi/APIcture/metrics.html
Evolution of API structure size metrics
paths and operations over time
paths
operations
Evolution of Size metrics
Parameters over time
parametered_operations
distinct_parameters
parameters_per_operations
used_parameters
Evolution of API methods
Methods over time
get
post
put
delete
patch
options
head
trace
Evolution of API Datamodel
Schemas and properties over time
schemas
properties
distinct_properties
Breaking Changes
Radar chart
response-required-property-removed
request-property-became-enum
request-property-type-changed
api-deprecated-sunset-parse
api-path-sunset-parse
request-property-became-required
Methods affected by breaking changes

Fig. 4. apict metrics visualization in `metrics.html`

- `viz-<openapi api file name>.html`: a single HTML page that includes all the previous interactive visualizations, in addition to a header showing history related metadata (Figure 5).

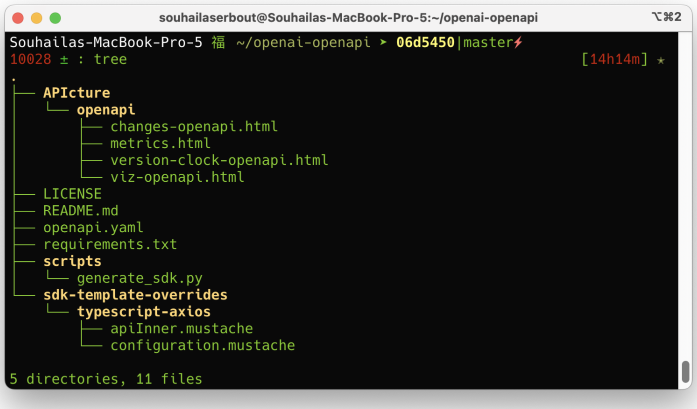

file:///Users/souhailaserbout/openai-openapi/APIcture/openapi/viz-openapi.html
API Version Clock
API Changes

API Meta data
Source: https://github.com/openai/openai-openapi.git
OAS File: openapi.yaml
API Title: OpenAI API [2022-10-19 18:53:13 -0700]

API Commits and Versions

Unique Versions	6
API Total Commits	36
API First commit	2022-01-19 09:44:49 -0800
API Last Commit	2023-08-12 18:07:11 -0400

Fig. 5. apict output evolution visualizations generated in `viz-openapi.html`



```
souhailaserbout@Souhailas-MacBook-Pro-5:~/openai-openapi                ⌥⌘2
Souhailas-MacBook-Pro-5 福 ~/openai-openapi ➤ 06d5450|master⚡
10028 ± : tree                                                   [14h14m] ★
.
├── APIcture
│   └── openapi
│       ├── changes-openapi.html
│       ├── metrics.html
│       ├── version-clock-openapi.html
│       └── viz-openapi.html
├── LICENSE
├── README.md
├── openapi.yaml
├── requirements.txt
├── scripts
│   └── generate_sdk.py
└── sdk-template-overrides
    └── typescript-axios
        ├── apiInner.mustache
        └── configuration.mustache

5 directories, 11 files
```

Fig. 6. `apict` command outputs

When executing the `apict` command without specifying any additional options, it initiates the process of generating an evolution report directly within the terminal (Figure 7). Furthermore, for users seeking to access this report independently of the complete visualization generation process, the `report` subcommand can be employed.

```
apict report -r openai-openapi
```

Rather than being limited to using only the overarching `apict` command, users have the flexibility to employ focused subcommands. Each of these subcommands generates a specific output individually, allowing for a more tailored approach to visualization creation. In addition to this, the `-fast` option is provided to optimize the time taken in the generation process, particularly when users are exclusively interested in obtaining a particular output. This approach streamlines the generation time, making the process more efficient and relevant to the specific visualization needs of the user.

2.4 Other supported cases

In scenarios where no OpenAPI file is detected within the repository (Figure 8), APIcture employs a distinct approach for Express.js projects. It systematically generates a corresponding OpenAPI specification from the project's codebase for each commit existing in the repository's history. Subsequently, APIcture selects the specifications that exhibit differences from the specification of the preceding commit. This generation process leverages ExpressO[1] [3] a CLI tool designed to validly generate OpenAPI specifications from expressjs code. The generated specifications are then utilized by APIcture to generate the intended visualizations.

In the case where the project's dependencies are not already installed, select `(x) No` then rerun the `apict` command.

It is important to note that the showcased examples within our published gallery exclusively originate from projects that feature an accessible OpenAPI specification within the repository. In this version of APIcture, the effective generation of visualizations through Express.js code hinges on the capability of ExpressO to construct a specification from the underlying codebase.

[3]https://www.npmjs.com/package/expresso-api

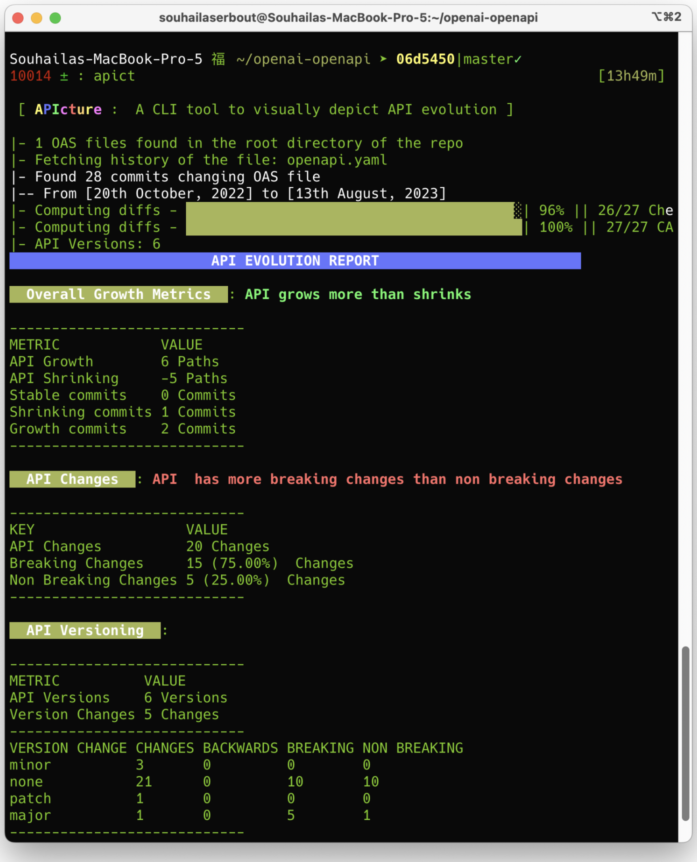

```
souhailaserbout@Souhailas-MacBook-Pro-5:~/openai-openapi                    ⌥⌘2

Souhailas-MacBook-Pro-5 福 ~/openai-openapi ➤ 06d5450|master✓
10014 ± : apict                                                     [13h49m]

 [ APIcture :  A CLI tool to visually depict API evolution ]

|- 1 OAS files found in the root directory of the repo
|- Fetching history of the file: openapi.yaml
|- Found 28 commits changing OAS file
|-- From [20th October, 2022] to [13th August, 2023]
|- Computing diffs -                                        ░| 96% || 26/27 Che
|- Computing diffs -                                         | 100% || 27/27 CA
|- API Versions: 6
                        API EVOLUTION REPORT

 Overall Growth Metrics : API grows more than shrinks

----------------------------
METRIC            VALUE
API Growth        6 Paths
API Shrinking     -5 Paths
Stable commits    0 Commits
Shrinking commits 1 Commits
Growth commits    2 Commits
----------------------------

 API Changes : API  has more breaking changes than non breaking changes

----------------------------
KEY                  VALUE
API Changes          20 Changes
Breaking Changes     15 (75.00%)  Changes
Non Breaking Changes 5 (25.00%)  Changes
----------------------------

 API Versioning :

----------------------------
METRIC          VALUE
API Versions    6 Versions
Version Changes 5 Changes
----------------------------
VERSION CHANGE CHANGES BACKWARDS BREAKING NON BREAKING
minor          3       0         0        0
none           21      0         10       10
patch          1       0         0        0
major          1       0         5        1
----------------------------
```

Fig. 7. apict terminal prompt

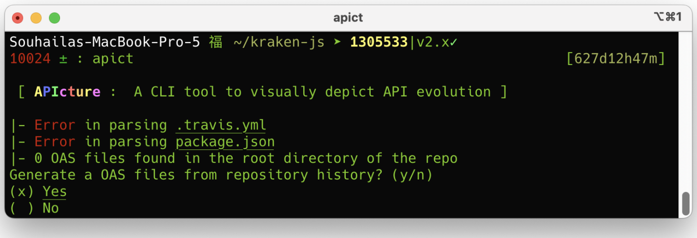

```
                                  apict                                     ⌥⌘1
Souhailas-MacBook-Pro-5 福 ~/kraken-js ➤ 1305533|v2.x✓
10024 ± : apict                                                  [627d12h47m]

 [ APIcture :  A CLI tool to visually depict API evolution ]

|- Error in parsing .travis.yml
|- Error in parsing package.json
|- 0 OAS files found in the root directory of the repo
Generate a OAS files from repository history? (y/n)
(x) Yes
( ) No
```

Fig. 8. APIcture in the case where no OpenAPI file is found in the project (run on the Kraken.js project)

3 CONTACT

We encourage the Artifact Evaluation Committee to reach out to the authors for any inquiries, feedback, or support related to the evaluation process. We are committed to providing assistance and addressing any questions that may arise during the evaluation.

Issues can be also added to the public repository of APIcture: https://github.com/souhailaS/APIcture. We are currently actively enhancing and maintaining it.

REFERENCES

[1] Souhaila Serbout, Alessandro Romanelli, and Cesare Pautasso. Expresso: From express.js implementation code to openapi interface descriptions. In Thais Batista, Tomáš Bureš, Claudia Raibulet, and Henry Muccini, editors, *Software Architecture. ECSA 2022 Tracks and Workshops*, pages 29–44, Cham, 2023. Springer International Publishing. ISBN 978-3-031-36889-9.