Mine Information from Archived Content

Mentors : Stefano Zacchirolli, Valentin Lorentz, Kumar Shivendu

Abstract

Software Heritage is a far-reaching Open Source-Research project that, at its core, archives software source code, hereby keeping it from becoming extinct. As a part of this, Software Heritage's indexer extracts from source code repositories, certain metadata ranging from simple information (eg. project name or hosting place) to more substantial ones like the entity behind the project, its license, etc. (paper-1). According to Software Heritage, metadata is the information it collects and extracts that provides additional information on source code (ref). This project aims to contribute to the sharing phase (paper-2) where help is needed to organize contents to build efficient indexing and querying mechanisms and to develop applications for specific domains.

What do I hope to Achieve?

- Indexing variants of regular metadata files :
    a. Add to intrinsic metadata files to be indexed: AUTHORS and CONTRIBUTORS

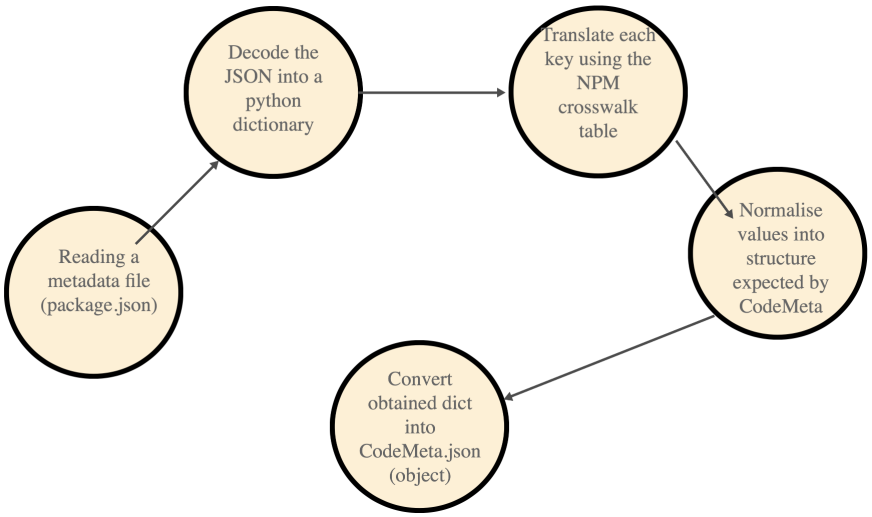
- Mining additional intrinsic metadata from archived content. This is by extending coverage of supported metadata to the following package managers (enclosed in parenthesis are the metadata files' names):
    a. Homebrew (7.52k packages) -> (my-new-cask-name.rb)
    b. Cocoapods (87.4k packages) -> (project-name.podspec)
    c. Go (369k packages) -> (go.mod)
    d. Packagist (357k packages) -> (composer.json)
    e. Nuget (377k packages) -> (NuGet.config)
    f. Pub (32k packages) -> (pubspec.yaml)

- Mining extrinsic metadata like
    a. Stars of a GitHub repo
    b. List of contributors

About the Project

There are two parts of the project. In the first part, I will work on mining intrinsic metadata from archived repositories. The second part focuses on extrinsic metadata. This link contains a description of the metadata workflow and architecture.
The CodeMeta initiative's crosswalk table provides a framework for translating software metadata into a convenient and common vocabulary. Once some filename-based heuristics indicate a file to be a metadata file, it is fetched and translated into CodeMeta metadata (JSON) format. Shown in the figure below is an

example sequence that follows in obtaining metadata of NPM-managed projects in the archive.



Usefulness

Software Heritage has a search feature that was implemented as an earlier GSoC project. The search feature also allows searching by metadata (say by searching for a contributor, it should provide all the projects the person has contributed to). Mining more such metadata helps improve the search feature of the archive. This project also improves the current indexer, which skips some metadata files due to various issues (paper-3). After the project, the indexer will address all variants (eg. MarkDown, RST, HTML) of metadata files. Most interesting of all, a search query language that queries the entire archive is available for beta users. The more the metadata available at SWH-SEARCH's disposal, the better the results it shall provide.

Workplan

Before May 19, 2022 (Before the Projects are announced)

- Learn more about linked data technologies and ontologies (RDFa, JSON-LD, OWL)
- Better understanding the architecture of the Software Heritage project codebase with the help of mentors.
- Discuss with mentors the prospect of working on a research paper related to my project and plan the project accordingly.

May 20, 2022, to June 13, 2022 (Community Bonding Period)

- Keep in constant touch with mentors to get a clear understanding of my goals and approach toward the project.

- With the help of others in the community, I would like to better understand the project, and how it functions. This I plan on doing by solving tasks and bugs listed in the forge.
- I would also like to invite my connections to explore the project and help them contribute wherever they see fit.

Phase I (June 13, 2022, to July 25, 2022)

June 13 to June 27 (Week 1 & 2)
- Work on improving the indexer by enabling it to index variants of regular metadata files.
- Solve simple indexer-related issues.

June 27 to July 11 (Week 3 & 4)
- Work on extending coverage of supported metadata to the Homebrew package manager.

July 11 to July 25 (Week 5 & 6)
- Work on extending coverage of supported metadata to Cocoapods package manager.

Mid Evaluations: July 25, 2022, to July 29, 2022

Phase II (July 25, 2022, to September 4, 2022)

July 25 to August 8 (Week 7 & 8)
- Work on extending coverage of supported metadata to the Go package manager.

August 8 to August 22 (Week 9 & 10)
- Work on extending coverage of supported metadata to Packagist package manager.

August 22 to September 5 (Week 11 & 12)
- Buffer week for uncompleted work

OR

- Mine extrinsic information from respective Origins. For example Stars of a GitHub Repository, List of Contributors.

OR

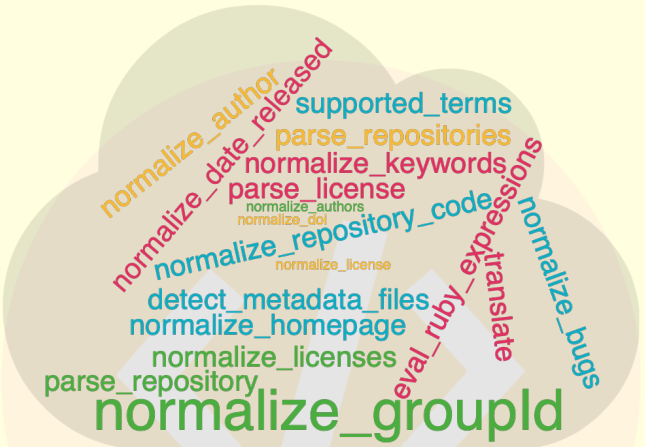
- Extend Supported metadata to the NuGet package manager.

Submissions Week (September 5, 2022, to September 12, 2022)
- Fixing bugs
- Cleaning the codebase
- Seeking mentor’s final reviews

Implementation Plan

For each of Homebrew, Cocoapods, Go and Packagist, a mapping needs to be created in the metadata_dictionary of the indexer. This mapping derives from one or more of the following classes: SingleFileMapping, DictMapping, or JsonMappin depending on whether the mapping takes as input a single file, a key-value store (dictionary), or JSON file, respectively.

Here are the supported methods for all the mapping classes currently available. Similar methods, based on the library being mapped, will be written for the above 4 package managers.



The cask-cookbook documentation contains all the metadata that can be extracted from Homebrew. When it comes to CocoaPods, the podfile is the file to target for metadata. The packages of Go package manager have the following structure.
For Packagist packages, there is the 'composer.json' file along with some other xml files(example) as given here. Finally, for Pub packages it'll be the pubspec.yaml file and for Nuget packages, I'll refer to the NuGet.config file of the source code.

While extracting extrinsic information about an origin is planned only if the earlier planned objectives are accomplished, here is a working plan for them. Usually, such extrinsic data isn't directly available (even through APIs of GitHub, GitLab etc.). Hence, third party APIs need to be developed. For example, here is a popular place to get the **GitHub** star history of any origin. As a continuation of this project, I could build an API to extract such content for Software Heritage's archive.

Future Aspects

- Work on a research paper focusing on the scale of things being dealt with in this project. The opportunity to study and analyze code functioning at a scale of more than 12 billion source files doesn't come every day.

- Possibly the research project could study the improvements to the search feature added by additional metadata that has been mined. Anticipating a lot of testing code to be written for the paper. Will use the pytest framework for this.

- Pursue solving the problems described in the following internship : https://wiki.softwareheritage.org/wiki/Expand_package_metadata_coverage_(internship)

Activity

Please find my activity on the project's forge, including my diffs, any merges, and other comments/discussions, here.

About Me

I am Satvik Vemuganti, a 2nd-year undergraduate student pursuing my Bachelor of Technology in Computer Science and Engineering from
IIT Bhilai, India. Fond of numbers from an early age, I chose to study the very machines that redefined Mathematics, as I grew older. As an aspiring researcher, I am fond of studying and experimenting with things for the sole satisfaction of the analysis that can be drawn from the results of these experiments. And so, I dream of pursuing higher studies after my Bachelor's degree to enable myself to work on the cutting edge of Computer Science research.

Somehow, the courses I have taken so far have led me to develop a deep fondness for machine learning and app development. My participation in Open Source events has instilled in me strong but passionate advocacy for Software that is Open Source. Software Heritage is an embodiment of Open Source software as it is built on and works on FOSS. This, to me, was an idea worth contributing towards and a dream worth working towards.

LinkedIn
GitHub
Portfolio
CV