Brief Introduction into ClearlyDefined Japan WG Tooling Subgroup

Hiroyuki FUKUCHI

ClearlyDefined

- ClearlyDefined
  - URL: https://docs.clearlydefined.io/
  - Parent Organization: Open Source Initiative

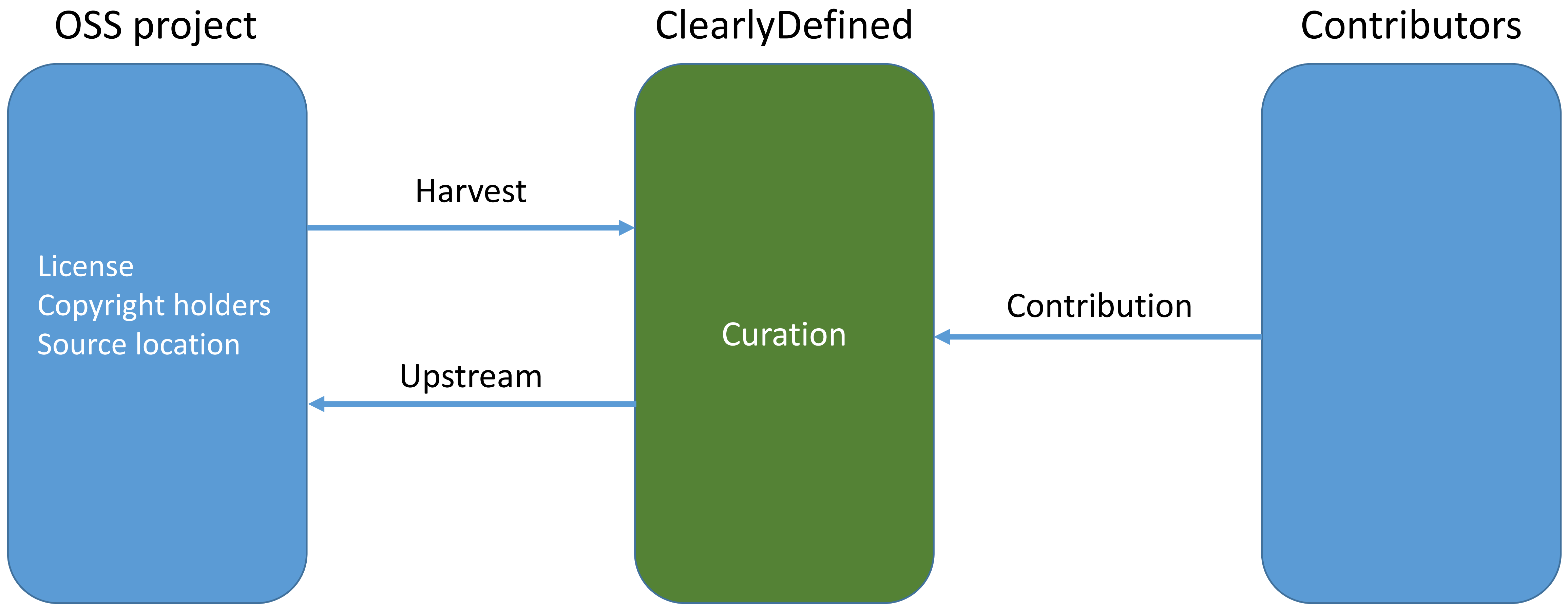
- Mission
  - To help FOSS projects thrive by being, well, clearly defined

- Goal
  - Raise awareness about this challenge within FOSS project teams
  - Automatically harvest data from projects
  - Make it easy for anyone to contribute missing information
  - Crowd-source the curation of these contributions
  - Feed curated contributions back to the original projects

Principles

- Neutral
  - The project carries no affiliation or company driven focus
- Open
  - The data, infrastructure, and processes are open to all
- Factual
  - All data is factual. No interpretation or assessment is made
- Upstream
  - Enable upstream projects as much as possible
- Simple
  - Wherever possible the project will use the simple solution

Image of ClearlyDefined activity



Scope

- License (declared and observed)
- Copyright holders
- Source location (including revision/commit)

ClearlyDefined

ClearlyDescribed

ClearlyLicensed

ClearlySecure

Example of Curation #1

```
{
	"described": {
		"sourceLocation": {
			"type": "git",
			"provider": "github",
			"url": "https://github.com/microsoft/redie",
			"revision": "194269b5b7010ad6f8dc4ef608c88128615031ca"
		}
	},
	"licensed": {
		"license": {
			"expression": "MIT"
		}
	}
}
```

Example of Curation #2

```
{
        "described": {
                "sourceLocation": {
                        "type": "git",
                        "provider": "github",
                        "url": "https://github.com/microsoft/redie",
                        "revision": "194269b5b7010ad6f8dc4ef608c88128615031ca"
                },
                "projectWebsite": "https://github.com/microsoft/redie",
                "issueTracker": "https://github.com/microsoft/redie/issues"
        },
        "licensed": {
                "license": {
                        "expression": "MIT"
                }
        }
}
```

Described (Source code location)

- sourceLocation
  - type
  - provider
  - url
  - revision
  - path
- issueTracker
- projectWebsite
- releaseDate
  - The ISO8601 date when the component was released.

Facet (Grouping of files)

- core
  - The files that go into making the release of the component. Note that the core facet is not explicitly defined. Rather, it is made up of whatever is not in any other facet. So, by default, all files are in the core facet unless otherwise specified.
- data
  - The files included in any data distribution of the component.
- dev
  - Files primarily used at development time (e.g., build utilities) and not distributed with the component
- docs
  - Documentation files. Docs may be included with the executable component or separately or not at all.
- examples
  - Like docs, examples may be included in the main component release or separately.
- tests
  - Test files may include code, data and other artifacts.

Licensed

- declared
  - The SPDX license expression that was declared to cover the component in general. For example, the value of the license property in an NPM or the license declared in the LICENSE file of a Git repo.
- files
  - Total number of files related to the facet.
- discovered
  - License information found in facet files. In particular, the following properties:
- expressions
  - The set of unique SPDX license expressions found across all files in the facet.
- unknown
  - The number of files in the facet that have no discernable license information.
- attribution
  - Information related to how attribution should be done for the files in the facet. Typically this equates to a list of copyright holders but projects vary as to how they want attribution to be done.
- parties
  - The set of unique entities that are to receive attribution (e.g., copyright holders)
- unknown
  - The number of files in the facet that have no discernable attribution information.

Secure

- Future work

Data

- type – the type of the component.
  - For example, npm, git, nuget, maven, …
- provider – where the component can be found.
  - Examples include npmjs, mavencentral, github, nuget, …
- namespace – many component systems have namespaces.
  - GitHub orgs, NPM namespace, Maven group id,.
- name – the name of the component.
- revision – components typically have some differentiator like a version or commit id.
- pr – literally the string pr.
- number – the GitHub PR number to apply to the existing harvested and curated data.

Future efforts

- Security
  - facilitating the reporting and tracking of vulnerabilities in projects
- Accessibility
  - Characteristics and analysis of a project’s support of accessibility related technology and concerns
- Project data
  - Governance model, principals, issue tracking, discussion forums, …

Community

- You can find the community at:
  - GitHub – github.com/clearlydefined
  - Discord – discord.gg/wEzHJku
  - Twitter – twitter.com/clearlydefd
  - Email – clearlydefined@googlegroups.com