DSpace Usage Statistics Collector Documentation

Release 0.1.2

LA Referencia

Aug 25, 2020

Contents:

CHAPTER 1

DSpace Usage Stats Collector

A python agent for sending DSpace usage statistics events to Matomo/OpenAIRE.

- Free software: GNU General Public License v3
- Documentation: https://dspace-stats-collector.readthedocs.io.

Implementation of a lightweight, easy-to-deploy, read-only alternative for a DSpace usage data collector compatible with Matomo and OpenAire usage statistics infrastructure. It sends usage data from individual repositories to an external regional aggregator by issuing read-only queries to the out-of-the-box DSpace Solr statistics subsystem.

A regional usage statistics service allows the sharing of data on item access across repositories, e-journals and CRIS systems in order to support evaluation, management and reporting. The success of this kind of service depends on installing a collector component in every repository, so one of the main requirements was to provide a user-friendly, non-invasive and reliable deploying process for repository managers.

This development is part of LA Referencia´s tasks in OpenAIRE Advance project, aimed to build a pilot on usage data exchange between Latin America and Europe open science infrastructures.

The design and the development of this usage data collector agent have been based on the following fundamental principles:

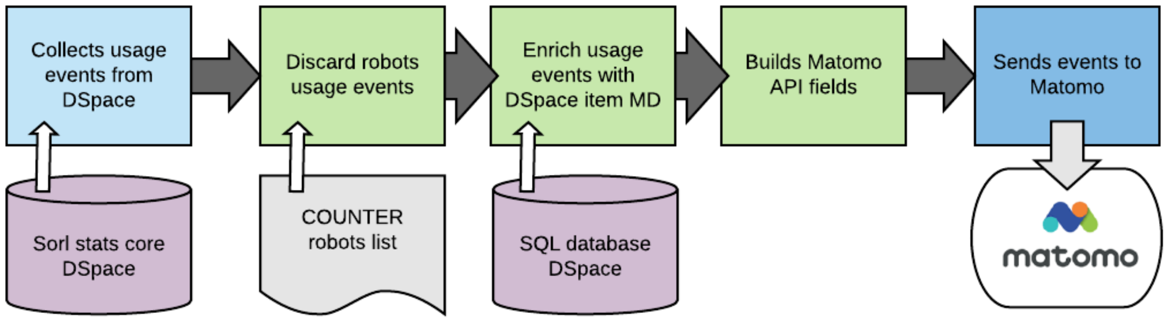
- open-source, collaborative development
- straightforward installation procedure for non-expert Linux users without root or superuser privileges
- capable of running in a sandbox without the need for installing system-wide packages in the host system
- light-weight and preserving system stability and performance
- fully compatible with OpenAIRE Usage Statistics Service [1]
- adaptable to other software platforms and aggregator services

1.1 Implementation highlights

The solution is based on a "pipe and filter" architecture with input, filter and output stages for events. This approach aims to factorize the problem in independent components, so more stages can be added/connected in the future, allowing to cover other software platforms.

In this first version of the agent, the following stages have been implemented for DSpace versions 4, 5 and 6, sending events to a Matomo instance, which is analysis platform used by the OpenAIRE [1]:

- DSpace Solr Statistics Input: an initial input component queries the internal DSpace Solr statistics core for new (later than a given/stored timestamp) usage events (item views/ item downloads). This initial event contains fields for timestamp, item id, user agent, IP address, among others
- COUNTER Robots Filter: this filter excludes events generated by internet robots and crawlers based on a list of user agent values provided by project COUNTER [3]
- DSpace Database Filter: this stage queries the internal DSpace relational database (currently only Postgres supported) for complementary item information which is not stored in the Solr core but is required by OpenAire specifications. This filter adds item title, bitstream filename and oai_identifier as event fields
- Matomo API Filter: this filter transforms previously gathered data into the set of parameters required by Matomo Tracking API [4]
- Matomo Sender Output: this filter buffers and sends batches of events into the regional tracker using the bulk tracking feature of Matomo HTTP Tracking API [4]



The resulting pipeline runs from the main collector script that stores the last successfully sent timestamp as a state for future calls.

1.2 Installation

https://dspace-stats-collector.readthedocs.io/en/latest/installation.html

1.3 Credits

This component is part of an alternative DSpace Usage Statistics collector strategy developed by LA Referencia / CONCYTEC (Perú) / IBICT (Brasil) / OpenAIRE as part of OpenAIRE Advance project - WP5 - Subtask 5.2.2. “Pilot common methods for usage statistics across Europe & Latin America”

1.4 References

[1] Schirrwagen, Jochen, Pierrakos, Dimitris, MacIntyre, Ross, Needham, Paul, Simeonov, Georgi, Príncipe, Pedro, & Dazy, André. (2017).

[2] OpenAIRE2020 - Usage Statistics Services - D8.5. doi: https://doi.org/10.5281/zenodo.1034164

[3] Python generators https://wiki.python.org/moin/Generators

[4] Project COUNTER https://www.projectcounter.org/

[5] Matomo tracking API, https://developer.matomo.org/api-reference/tracking-api

[6] DSpace Statistics https://wiki.lyrasis.org/display/DSDOC3x/DSpace+Statistics

CHAPTER 2

Installation

2.1 Installation - standalone user level installing (w/ python bundle)

The collector can be run manually or as a scheduled task using the CRON hosting system. A single bash installation script was developed for implementing a straightforward setup process. This bash script executes the following installation/configuration steps:

- download and install a free minimal Python environment (https://docs.conda.io/en/latest/miniconda.html) in the user home directory
- install required Python packages
- create the default configuration file
- download the latest COUNTER Robots file
- instruct the user to fill minimal information in the configuration file: the DSpace installation directory, the DSpace major version and the required credentials for sending events to a remote Matomo instance

After this simple installation process, the collector is ready to start working by collecting and sending usage data into the pre-configured remote Matomo instance. Also a command to install the collector script in the user CRONTAB is provided.

IMPORTANT: The instalation script and the dspace-stats-collector does not require superuser privileges and don´t install any software outside the CURRENT_USER_HOME/dspace-stats-collector. The collector script execute read only queries over dspace relational db and solr core. This tool doesn´t write or modify any dspace file, dspace db or solr core. It´s recommended, but not mandatory, execute the instalation script from de dspace user.

2.2 Installation steps:

1. Check if wget and cron are installed in the system.
2. Execute installation script from a plain user (ie: dspace) directly from github:

   wget -O - https://git.io/JvzBR | bash

3. Configure matomo site parameters provided in CURRENT_USER_HOME/dspace-stats-collector/config/default.properties
4. Execute CURRENT_USER_HOME/dspace-stasts-collector/bin/dspace-stats-collector -v -f YYYY-MM-DD (will collect and send events for the first time from YYYY-MM-DD)
5. Check if the collector is sending data to matomo instance by asking to your national node manager (do not execute the next step without this check)
6. Execute CURRENT_USER_HOME/dspace-stasts-collector/bin/dspace-stats-cronify (will install collector in user cron)
7. Check/ajust the user crontab (the instalation script adds an entry automatically in the user crontab, the collector runs every 60 min by default)

2.3 Update steps

1. Logged as the same user used in installation process run:

   wget -O - https://git.io/Jvz4q | bash

CHAPTER 3

Usage

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/lareferencia/dspace_stats_collector/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

Dspace usage stats collector could always use more documentation, whether as part of the official Dspace usage stats collector docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/lareferencia/dspace_stats_collector/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *dspace_stats_collector* for local development.

1. Fork the *dspace_stats_collector* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/dspace_stats_collector.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv dspace_stats_collector
$ cd dspace_stats_collector/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 dspace_stats_collector tests
$ python setup.py test or py.test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.7, 3.4, 3.5 and 3.6, and for PyPy. Check https://travis-ci.org/lareferencia/dspace_stats_collector/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ py.test tests.test_dspace_stats_collector
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bumpversion patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER 5

Credits

5.1 Development Lead

- LA Referencia <lareferencia.dev@gmail.com>

5.2 Contributors

- Lautaro Matas <lmatas@gmail.com> - LA Referencia
- César Olivares - CONCYTEC (Perú)
- Washington Riveiro - IBICT (Brasil)
- Vander Barreto - IBICT (Brasil)
- Rino Vargas - CONCYTEC (Perú)
- Guillermo Murillo - LA Referencia

CHAPTER 6

History

6.1 0.1.0 (2019-07-23)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search