Soft Dive Into GrimoireLab

CHAOSSCon North America - San Diego - Ago. 2019

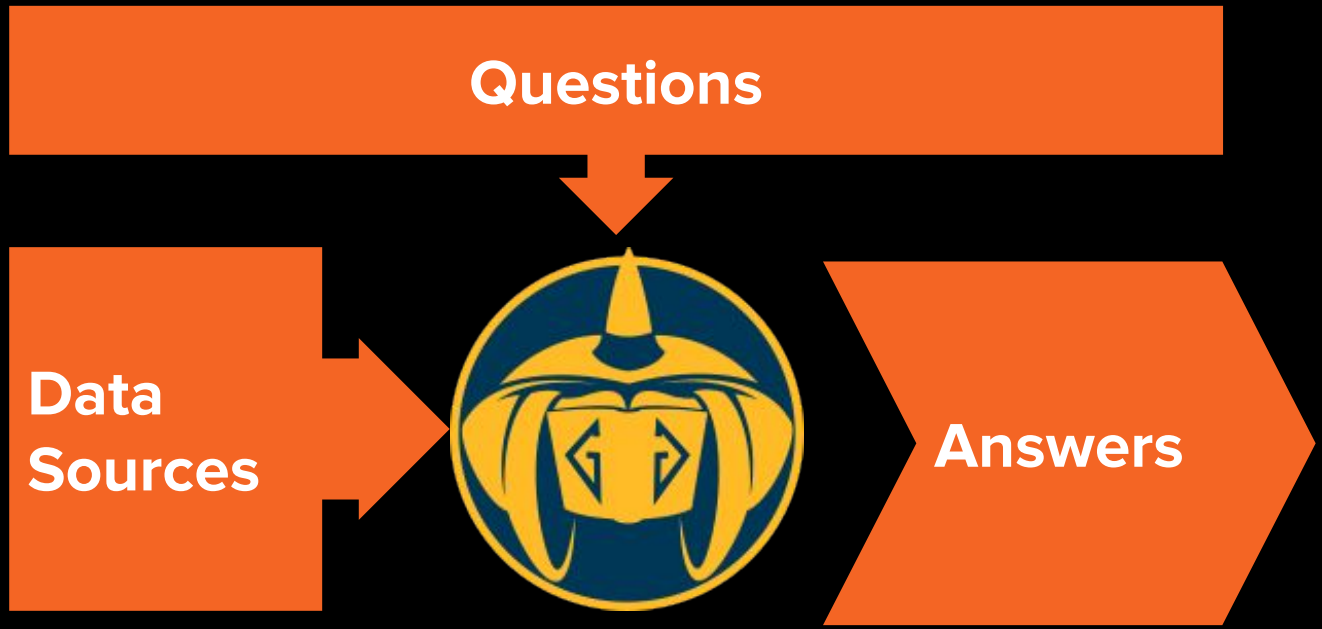
Manrique Lopez / Santiago Dueñas
@bitergia
jsmanrique@bitergia.com / sduenas@bitergia.com

Bitergia

GrimoireLab ... what is it?

The free, open source toolkit to answer your questions about the community and processes involved in software development

And, if it fails, or you know how to improve it, feel free to open an issue, send a pull request, or discuss about it in the mailing list ;-)



Bitergia

What are your questions?

Tip:

Don’t start with the questions.

Start with the goals

Think strategically!

Goals - Questions - Metrics

Goal

We want these projects to be community driven and not ruled by a single company

Questions

How many organizations are participating or have participated?

Is this a community driven project?

Metrics

Organizational diversity (CHAOSS Metric)

Gmail Factor: % of commits, and authors, from gmail.com accounts

CHAOSS Metrics

chaoss.community/metrics

Bitergia

Our own questions

How many Bitergians have contributed to CHAOSS working groups?

What is the responsiveness to CHAOSS community pull requests in GrimoireLab?

Is this a contributor driven community or a company driven community?

Workshop Materials

You need **git** and **Docker** installed

... and *some* GBs of RAM and HDD

Extra: Python might help if you wanna run some scripts locally

gitlab.com/jsmanrique/grimoirelab-workshop

```
~/$ git clone https://gitlab.com/Bitergia/lab/analytics-demo

~/$ cd analytics-demo

~/analytics-demo $
```

What do you want to analyze?

What's in a "project"?

Data sources

Repositories (*of data*)

Granted access to repositories

projects.json

```
{
  “grimoirelab”: {
      “meta”: {
          “title”: “...”,
          “logo_url”: “...”,
          “...”
      },
      “git”: [“...”, “...”, ..., “...”],
      “github”:  [“...”, “...”, ..., “...”],
      ...
  },
  ...
}
```

Bitergia

setup.cfg

```
...
[github]
api-token = <YOUR_API_TOKEN_HERE>
raw_index = github_demo_raw
enriched_index = github_demo_enriched
...
```

```
~/analytics-demo $ docker-compose up -d
```

1, 2, 3 ..., 100 (or *coffee time*)

Visit `http://<localhost_or_your-server-IP>:5601`

Bitergia

What's going on?

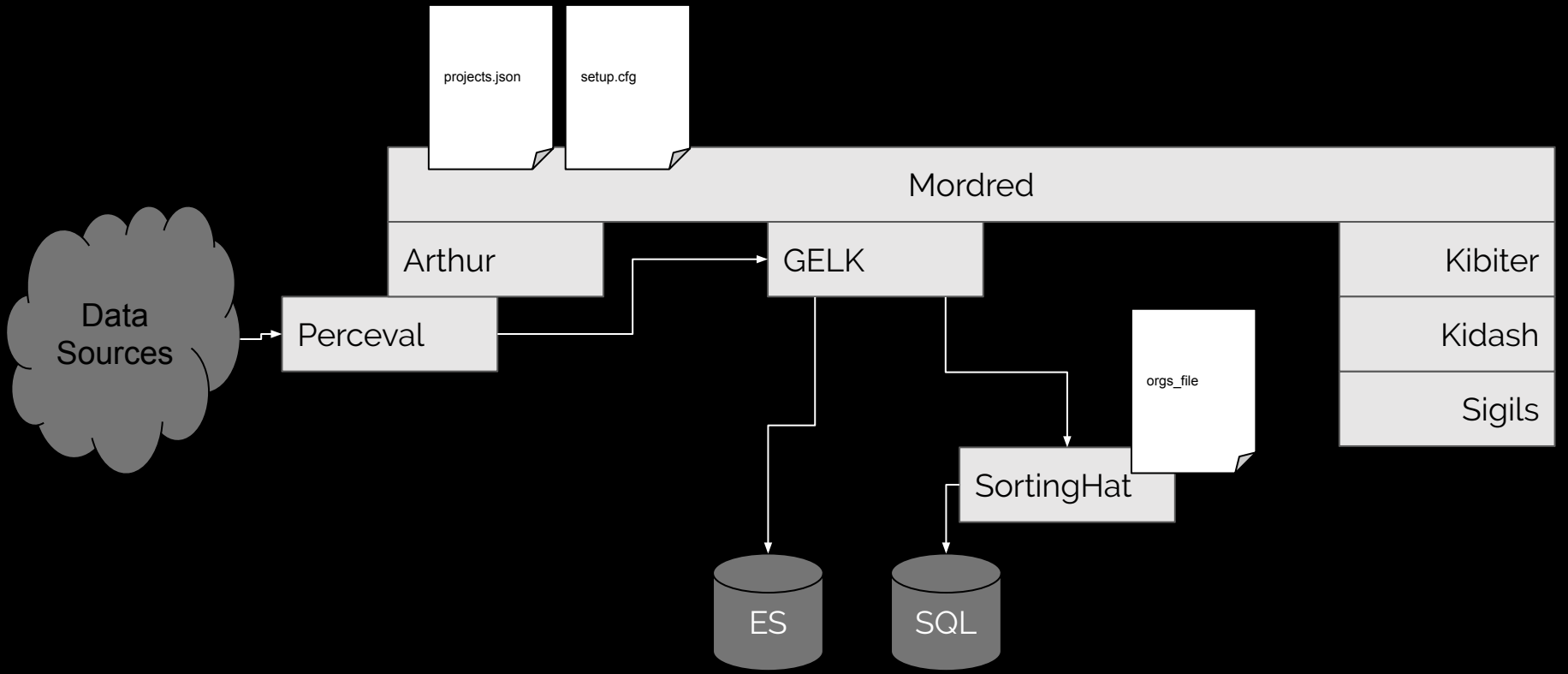
GrimoireLab Data Workflow

What's inside GrimoireLab?

Mordred		
Arthur	GELK	Kibiter
Perceval	SortingHat	Kidash
		Sigils
		Manuscripts

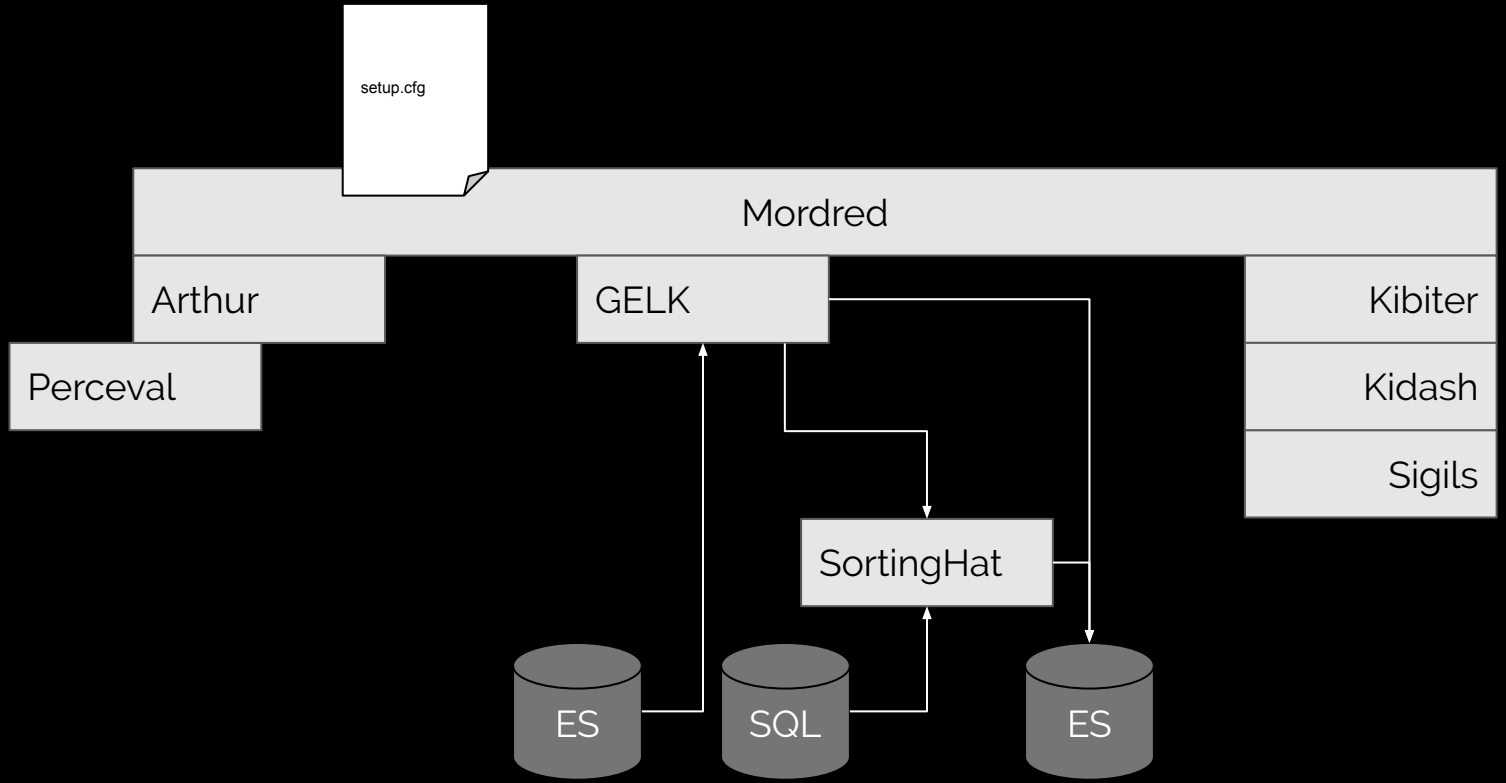
Bitergia

1st step: **Gathering data**


projects.json
setup.cfg
Mordred
Arthur
GELK
Kibiter
Data Sources
Perceval
Kidash
Sigils
orgs_file
SortingHat
ES
SQL

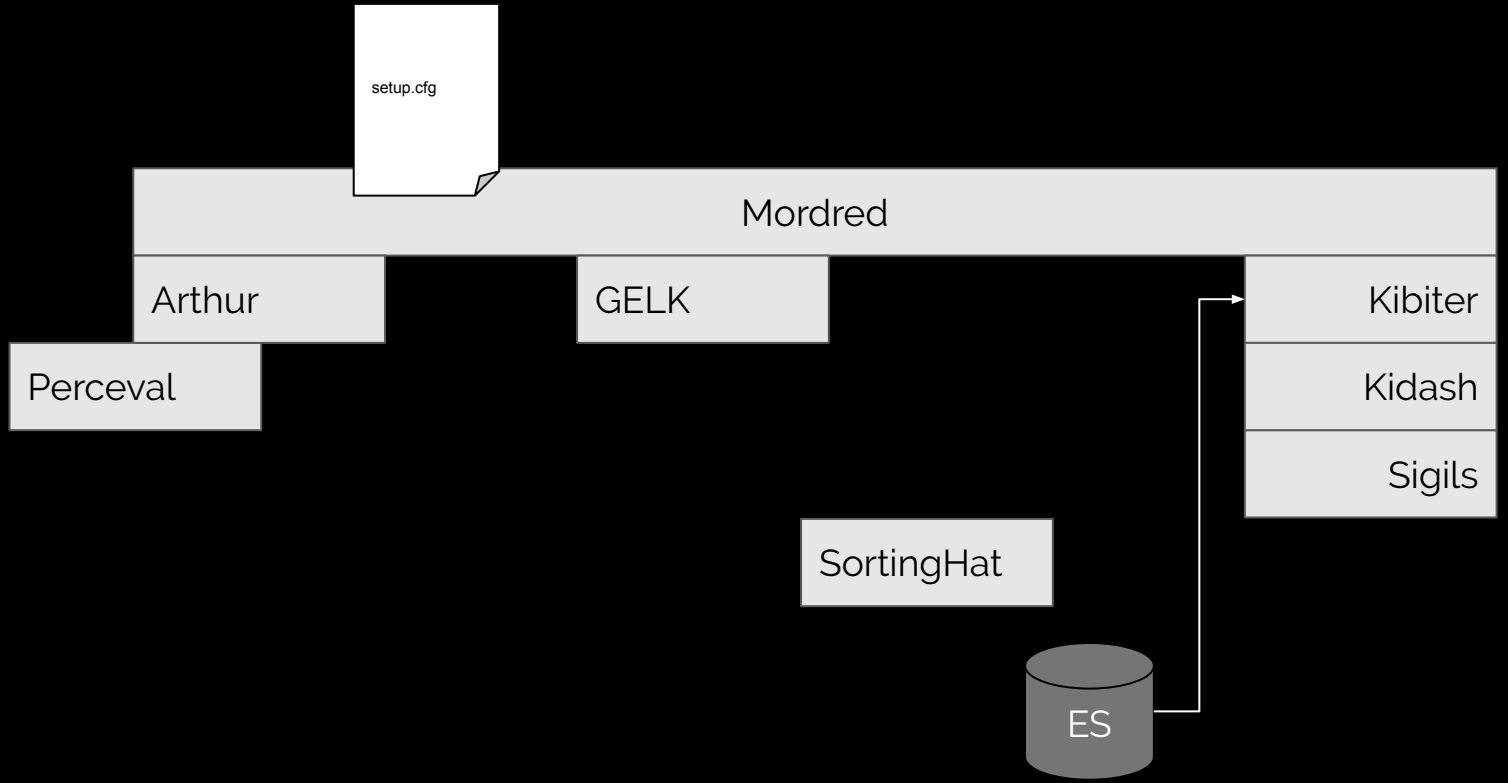
Bitergia

2nd step: **Data Enrichment**

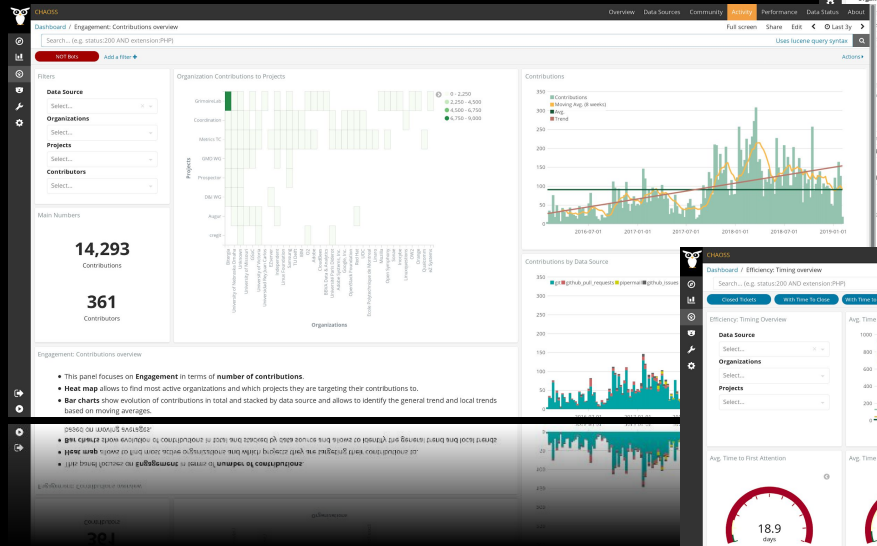

setup.cfg
Mordred
Arthur
Perceval
GELK
Kibiter
Kidash
Sigils
SortingHat
ES
SQL
ES

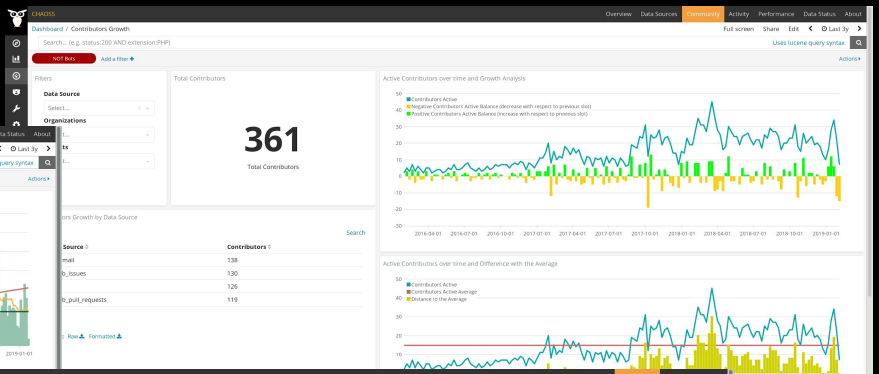
Bitergia

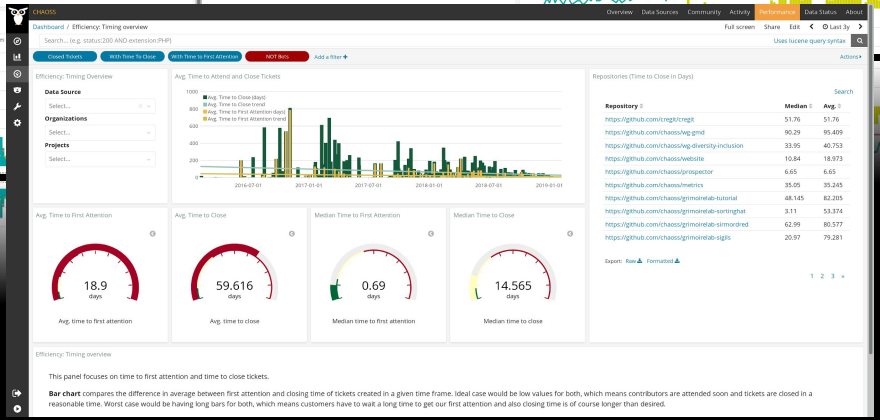
3rd step: **Data Consumption**


setup.cfg
Mordred
Arthur
GELK
Perceval
Kibiter
Kidash
Sigils
SortingHat
ES

Bitergia


CHAOSS
Dashboard / Engagement: Contributions overview
Main Numbers
14,293
Contributions
361
Contributors


CHAOSS
Dashboard / Contributors Growth
361
Total Contributors


CHAOSS
Dashboard / Efficiency: Timing overview
18.9
days
Avg. time to first attention
59.616
days
Avg. time to close
0.69
days
Median time to first attention
14.565
days
Median time to close

Bitergia

GrimoireLab Components

Note:
This is not a microservices architecture, yet :-)

Have fun playing with them!

Before starting

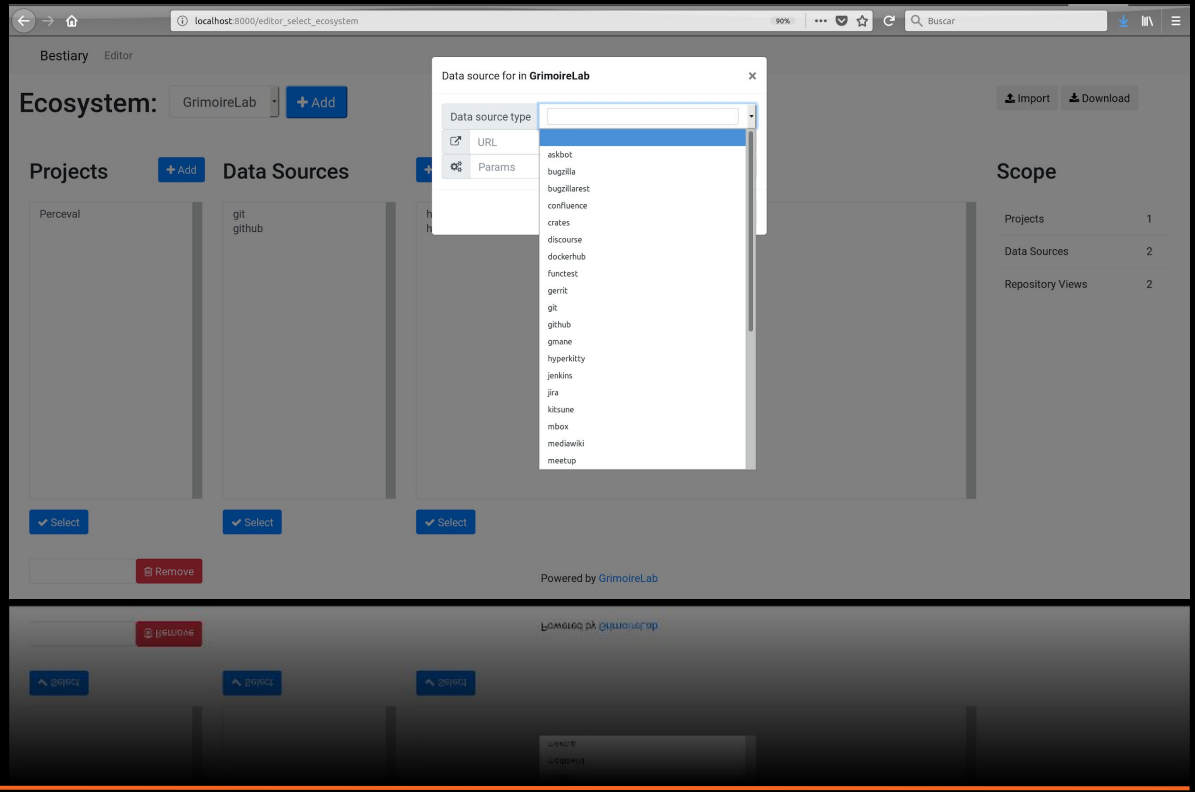
NOTE: To avoid installing components one by one, and to speed up training:

```
$ python3 -m venv /tmp/grimoirelab

$ source /tmp/grimoirelab/bin/activate

(grimoirelab) $ pip install --upgrade pip
setuptools wheel

...

(grimoirelab) $ pip install grimoirelab
```

Data Gathering

Bestiary

Work in progress: github.com/chaoss/grimoirelab-bestiary



Bitergia

Perceval

PyPI package:

```
$ pip3 install perceval
```

From sources:

```
$ git clone https://github.com/chaoss/grimoire-perceval.git
$ pip3 install -r requirements.txt
$ python3 setup.py install
```

Perceval

askbot	Fetch questions and answers from Askbot site
bugzilla	Fetch bugs from a Bugzilla server
bugzillarest	Fetch bugs from a Bugzilla server (>=5.0) using its REST API
confluence	Fetch contents from a Confluence server
discourse	Fetch posts from Discourse site
dockerhub	Fetch repository data from Docker Hub site
gerrit	Fetch reviews from a Gerrit server
git	Fetch commits from Git
github	Fetch issues, pull requests and repository information from GitHub
gitlab	Fetch issues, merge requests from GitLab
googlehits	Fetch hits from Google API
groupsio	Fetch messages from Groups.io
hyperkitty	Fetch messages from a HyperKitty archiver
jenkins	Fetch builds from a Jenkins server
jira	Fetch issues from JIRA issue tracker
launchpad	Fetch issues from Launchpad issue tracker
mattermost	Fetch posts from a Mattermost server
mbox	Fetch messages from MBox files
mediawiki	Fetch pages and revisions from a MediaWiki site
meetup	Fetch events from a Meetup group
nntp	Fetch articles from a NNTP news group
phabricator	Fetch tasks from a Phabricator site
pipermail	Fetch messages from a Pipermail archiver
redmine	Fetch issues from a Redmine server
rss	Fetch entries from a RSS feed server
slack	Fetch messages from a Slack channel
stackexchange	Fetch questions from StackExchange sites
supybot	Fetch messages from Supybot log files
telegram	Fetch messages from the Telegram server
twitter	Fetch tweets from the Twitter Search API

Perceval

From command line:

```
(perceval) $ perceval [-c <file>] [-g] \

             <backend> [<args>] |--help | --version
```

In your Python code:

```
…
from perceval.backends.core.<backend> import <Backend>
…
backend_repo = <Backend>(<params>)
for item in backend_repo.fetch():
…
```

Bitergia

Perceval

Write your own backends!

github.com/chaoss/grimoirelab-perceval/tree/master/perceval/backends/core

```
…
class <Backend>(Backend):
…
@metadata
def fetch(self):
    …
…
class <Backend>Client:
    …
class <Backend>Command(BackendCommand):
    …
```

Bitergia

Bitergia

GSoC: CHAOSS Metrics with Perceval

Aniruddha Karajgi

github.com/Polaris000/GSoC_19_Perceval_Implementations

Arthur (AKA KingArthur)

Work in progress: github.com/chaoss/grimoirelab-kingarthur

Scheduler for Perceval

Citadel

Work in progress: github.com/Bitergia/citadel

GSoC: Graal integration w GrimoireLab

Nishchith Shetty

github.com/inishchith/gsoc

Graal != GraalVM by Oracle

Graal leverages on the Git backend of Perceval and enhances it to set up ad-hoc source code analysis (code complexity, licensing, vulnerabilities, etc.)

More about Graal:

github.com/chaoss/grimoirelab-graal

blog.bitergia.com/2018/07/24/graal-the-quest-for-source-code-knowledge

Data Enrichment

Sorting Hat

Maintains an SQL database with identities that can be merged in the same unique identity.

For each unique identity, a profile can be defined: name, email, and other data.

Each unique identity can be related to one or more affiliations, for different time periods.

github.com/chaoss/grimoirelab-sortinghat

Sorting Hat

From command line:

```
(grimoirelab) $ sortinghat --help

...

(grimoirelab) $ sortinghat --host <DATABASE_IP>\
                           --user root \
                           --database <DATABASE_NAME> \
                           <COMMAND> <PARAMETERS>
```

Sorting Hat

In your Python code:

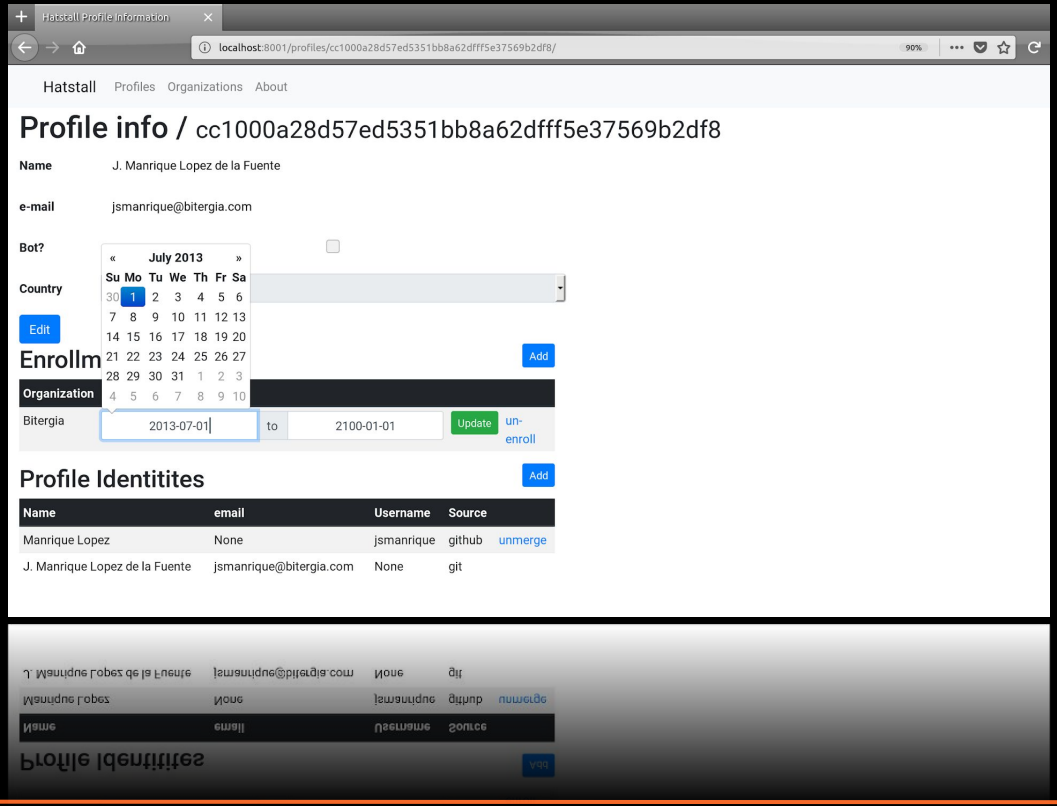
```
from sortinghat.db.database import Database
import sortinghat.api
from sortinghat.db.model import MIN_PERIOD_DATE,
MAX_PERIOD_DATE,\
    UniqueIdentity, Identity, Profile, Organization, Domain,\
    Country, Enrollment, MatchingBlacklist

sortinghat_db_connection = Database(user=<DB_USER>,\
                                    password=<DB_PASS>,\
                                    database=<DB_NAME>,\
                                    host=<DB_HOST>)

sortinghat.api.<COMMAND>(sortinghat_db_connection, <PARAMETERS>)
```

Bitergia

Bitergia

Hatstall

Work in progress: github.com/chaoss/grimoirelab-hatstall



Bitergia

Bitergia

SortingHat GraphQL API

Work in progress: github.com/chaoss/grimoirelab-sortinghat

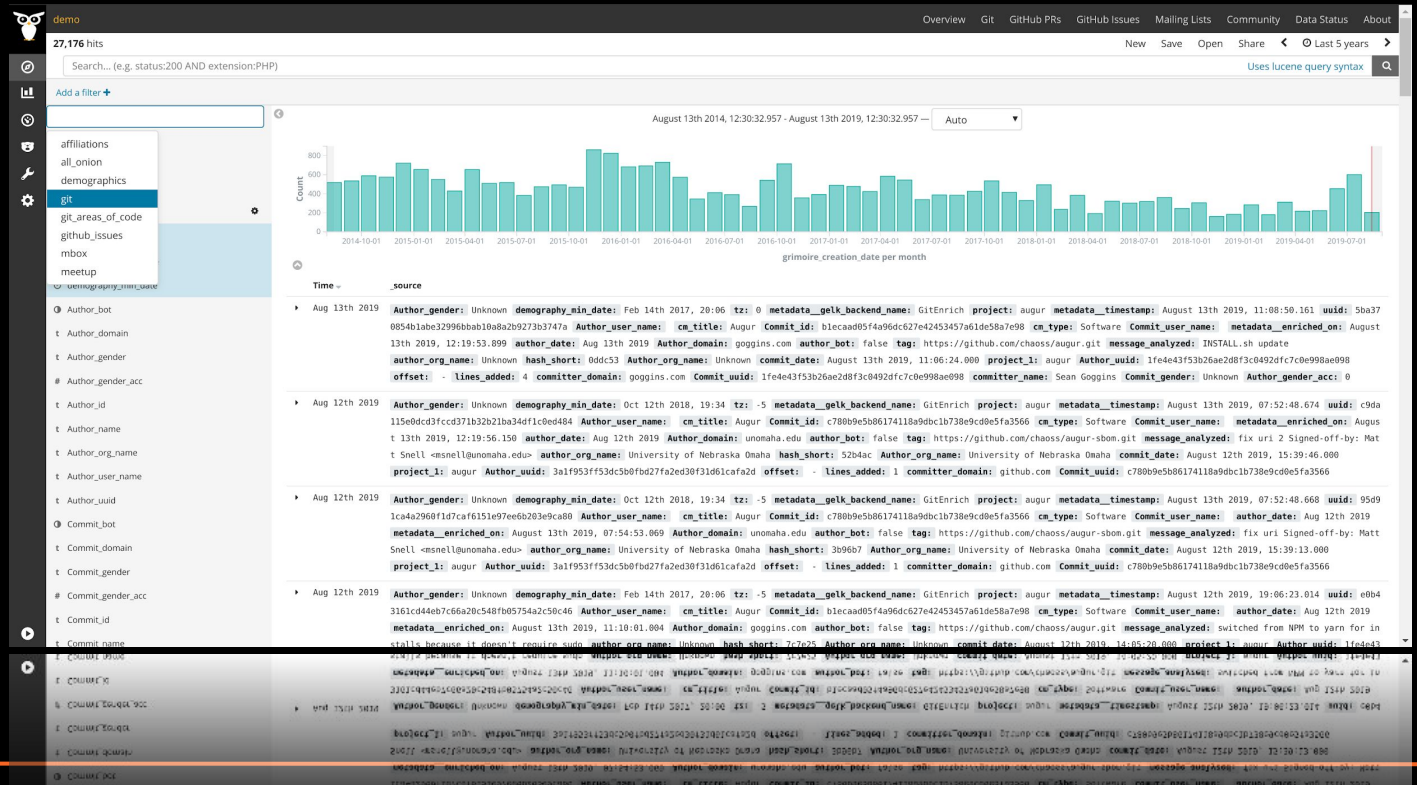
Raistlin

Work in progress: github.com/Bitergia/raistlin

Data Consumption

Data Schema

github.com/chaoss/grimoirelab-elk/tree/master/schema



Bitergia

Elasticsearch REST API

Python API

elasticsearch-py.readthedocs.io

elasticsearch-dsl.readthedocs.io

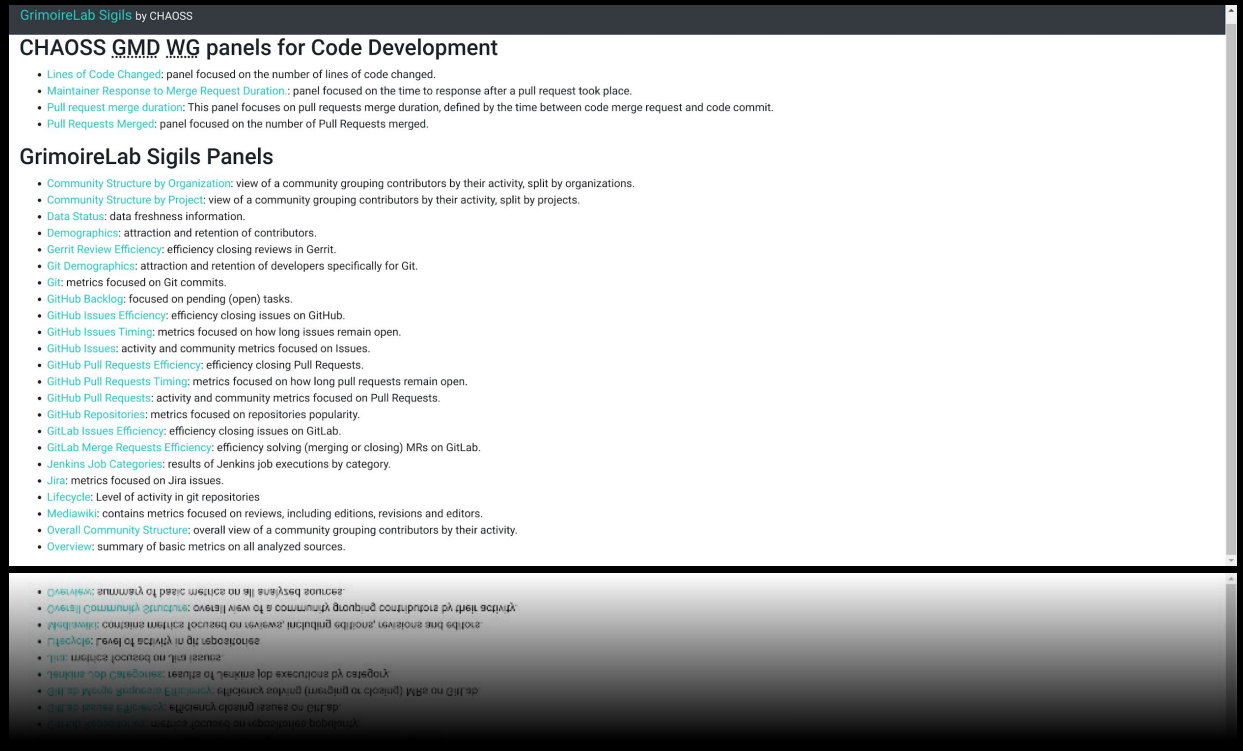
Javascript API

www.elastic.co/guide/en/elasticsearch/client/javascript-api/current/index.html

Bitergia

Sigils

chaoss.github.io/grimoirelab-sigils



GrimoireLab Sigils by CHAOSS

CHAOSS GMD WG panels for Code Development

- Lines of Code Changed: panel focused on the number of lines of code changed.
- Maintainer Response to Merge Request Duration.: panel focused on the time to response after a pull request took place.
- Pull request merge duration: This panel focuses on pull requests merge duration, defined by the time between code merge request and code commit.
- Pull Requests Merged: panel focused on the number of Pull Requests merged.

GrimoireLab Sigils Panels

- Community Structure by Organization: view of a community grouping contributors by their activity, split by organizations.
- Community Structure by Project: view of a community grouping contributors by their activity, split by projects.
- Data Status: data freshness information.
- Demographics: attraction and retention of contributors.
- Gerrit Review Efficiency: efficiency closing reviews in Gerrit.
- Git Demographics: attraction and retention of developers specifically for Git.
- Git: metrics focused on Git commits.
- GitHub Backlog: focused on pending (open) tasks.
- GitHub Issues Efficiency: efficiency closing issues on GitHub.
- GitHub Issues Timing: metrics focused on how long issues remain open.
- GitHub Issues: activity and community metrics focused on Issues.
- GitHub Pull Requests Efficiency: efficiency closing Pull Requests.
- GitHub Pull Requests Timing: metrics focused on how long pull requests remain open.
- GitHub Pull Requests: activity and community metrics focused on Pull Requests.
- GitHub Repositories: metrics focused on repositories popularity.
- GitLab Issues Efficiency: efficiency closing issues on GitLab.
- GitLab Merge Requests Efficiency: efficiency solving (merging or closing) MRs on GitLab.
- Jenkins Job Categories: results of Jenkins job executions by category.
- Jira: metrics focused on Jira issues.
- Lifecycle: Level of activity in git repositories
- Mediawiki: contains metrics focused on reviews, including editions, revisions and editors.
- Overall Community Structure: overall view of a community grouping contributors by their activity.
- Overview: summary of basic metrics on all analyzed sources.

Bitergia

Kidash

Tool to export & import GrimoireLab dashboards

```
(grimoirelab) $ kidash --help

...

(grimoirelab) $ kidash -e <Elasticsearch_IP> --list
```

Archimedes

Work in progress: github.com/Bitergia/archimedes

Opendistro for Elasticsearch

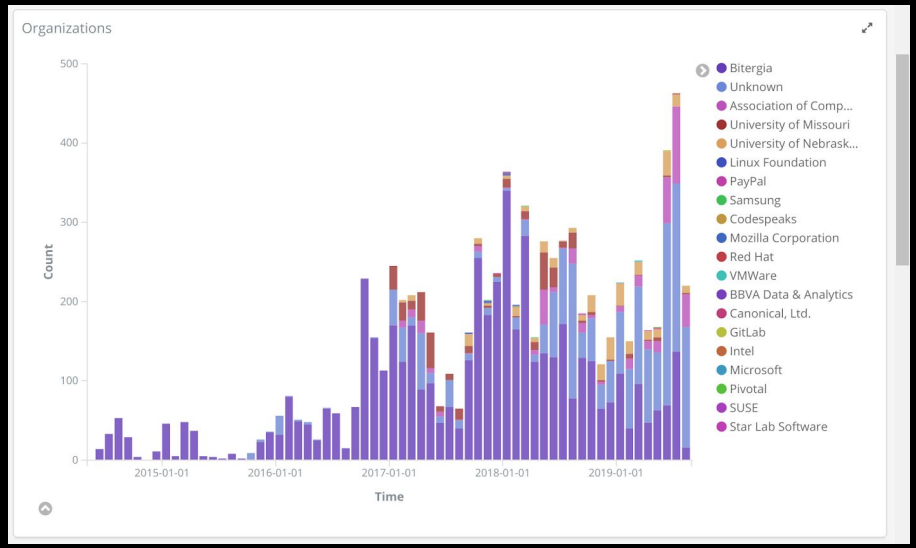
Being tested in alpha.cauldron.io

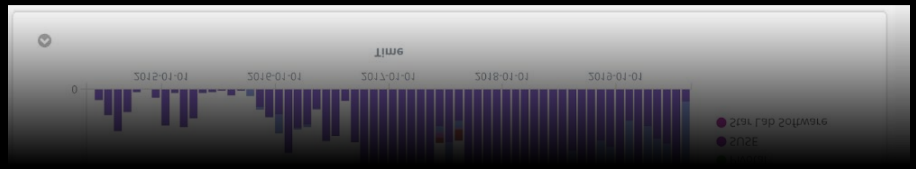
More about that later ;-)

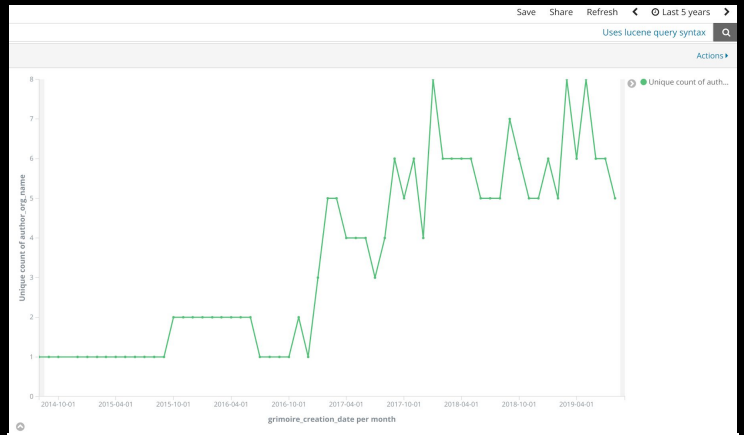
Let’s answer questions

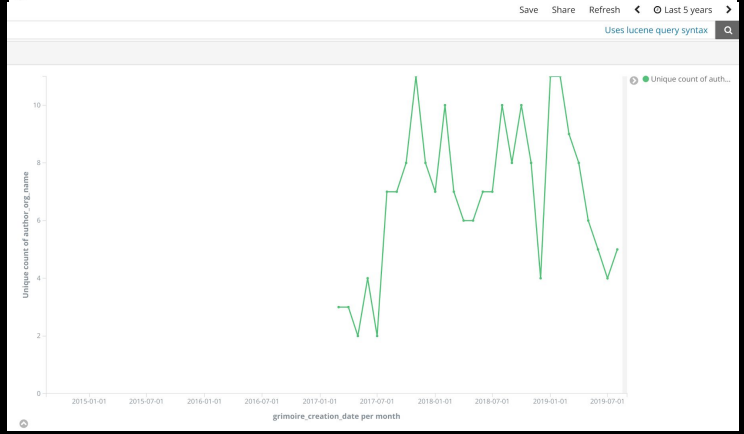
Is this a contributor driven community or a company driven community?

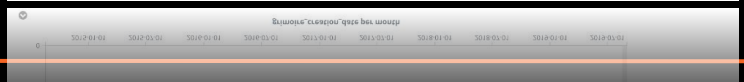
Organizational Diversity





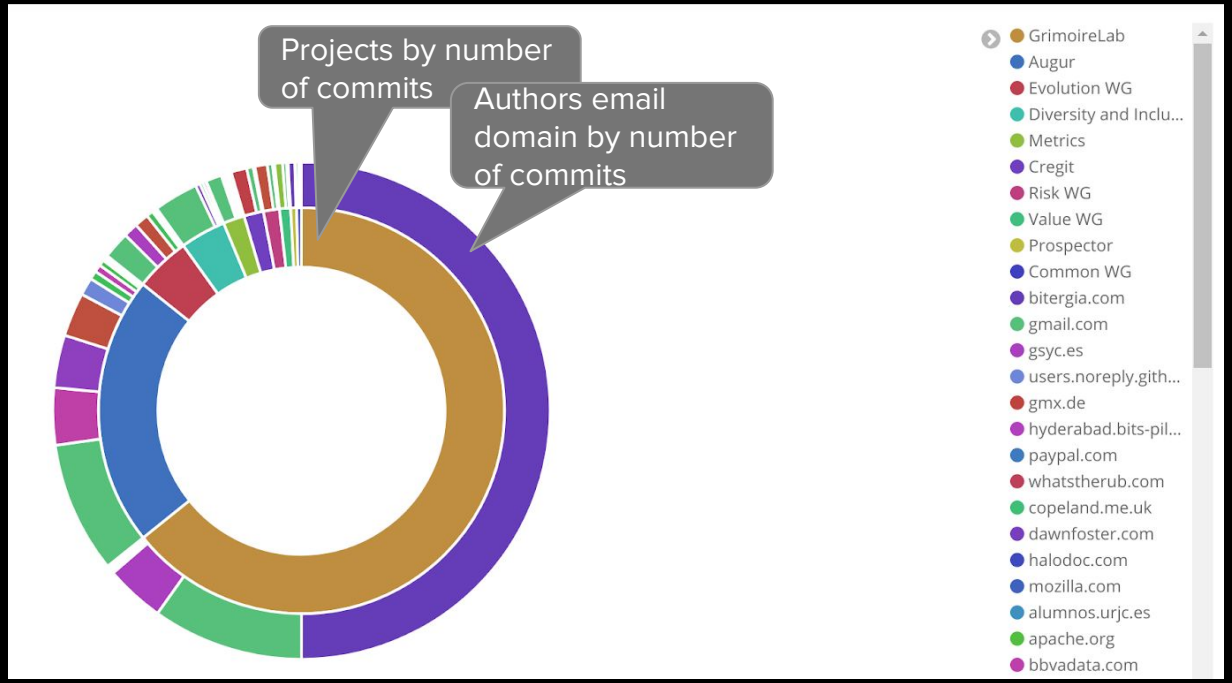






Bitergia

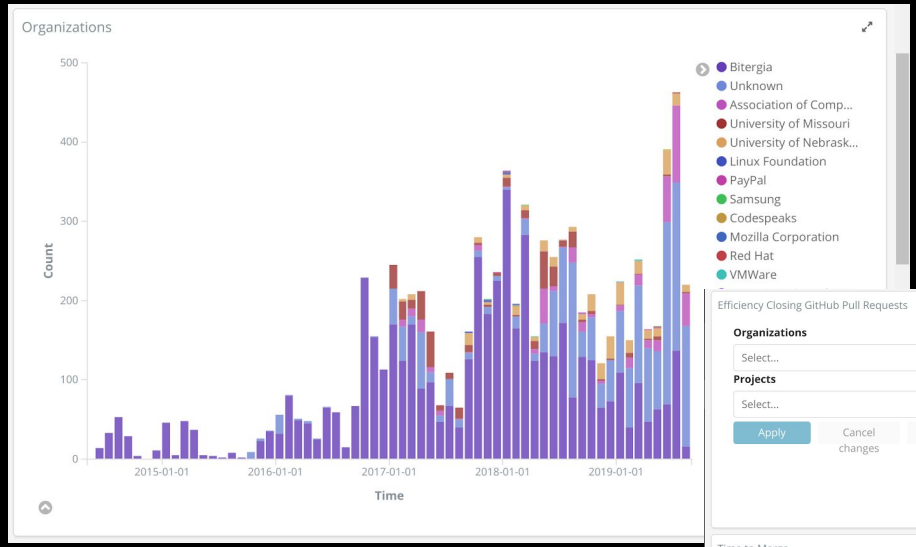
GMail Factor

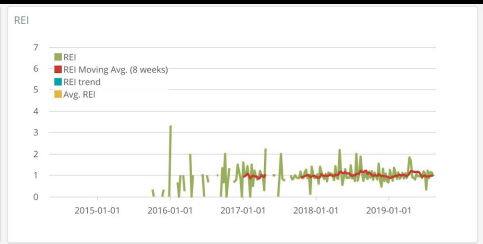


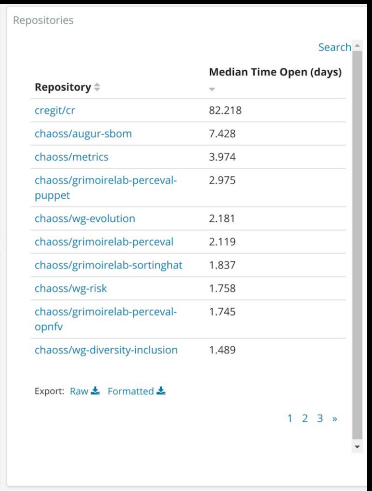
Bitergia

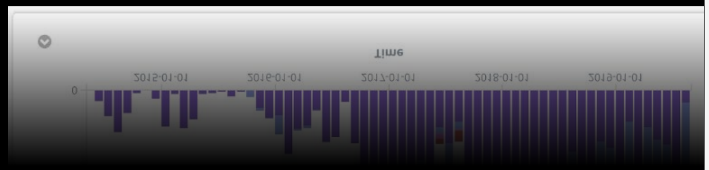
CHAOSS Metrics

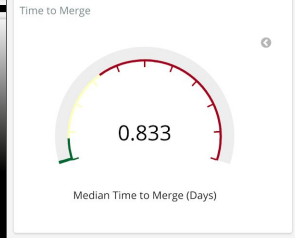
Evolution - Code Development

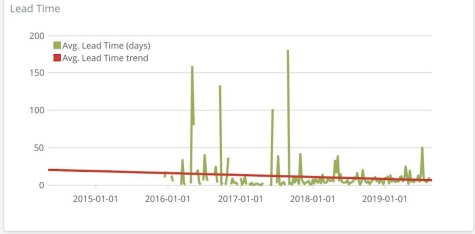


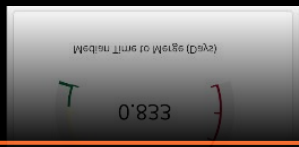


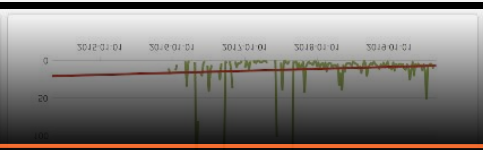


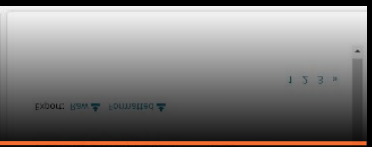








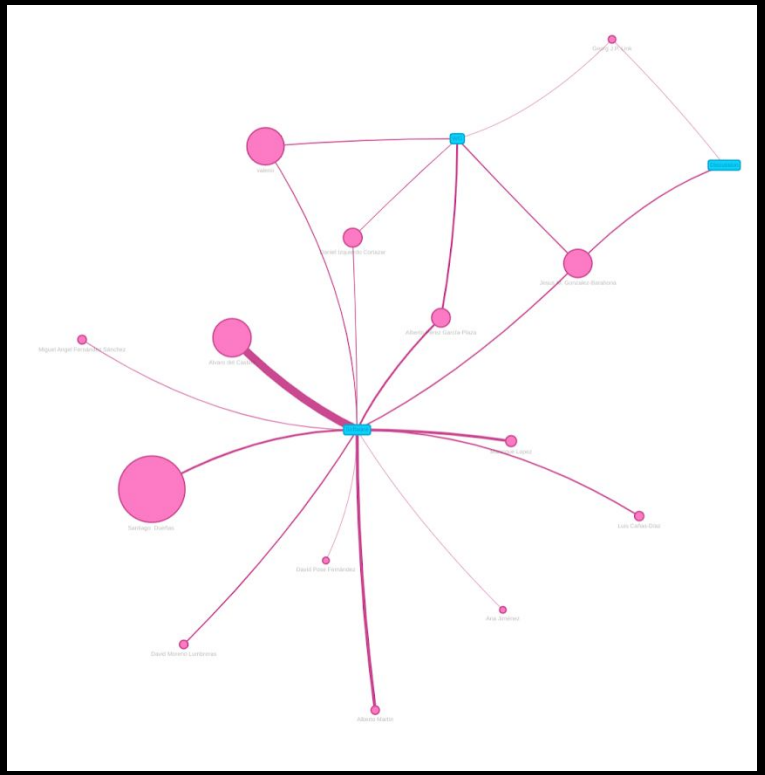


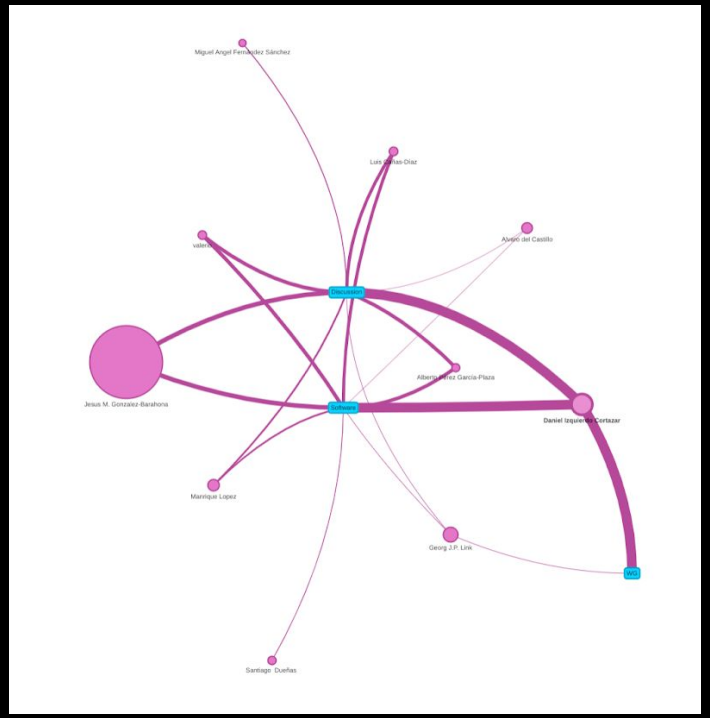


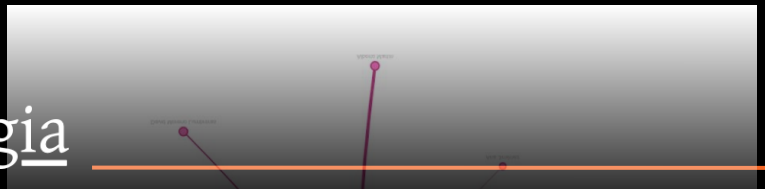
Bitergia

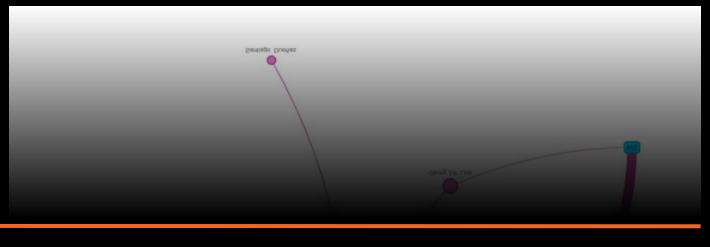
How many Bitergians have contributed to CHAOSS working groups?

Network view





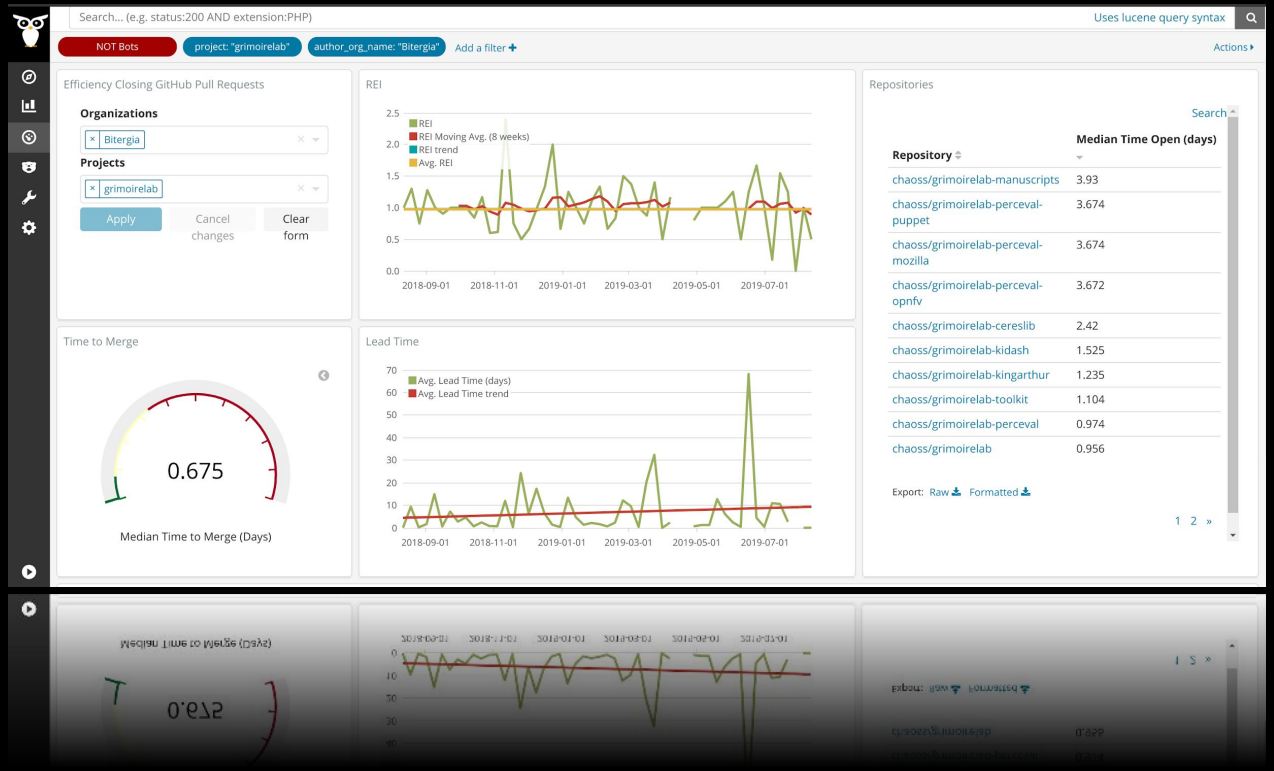




Bitergia

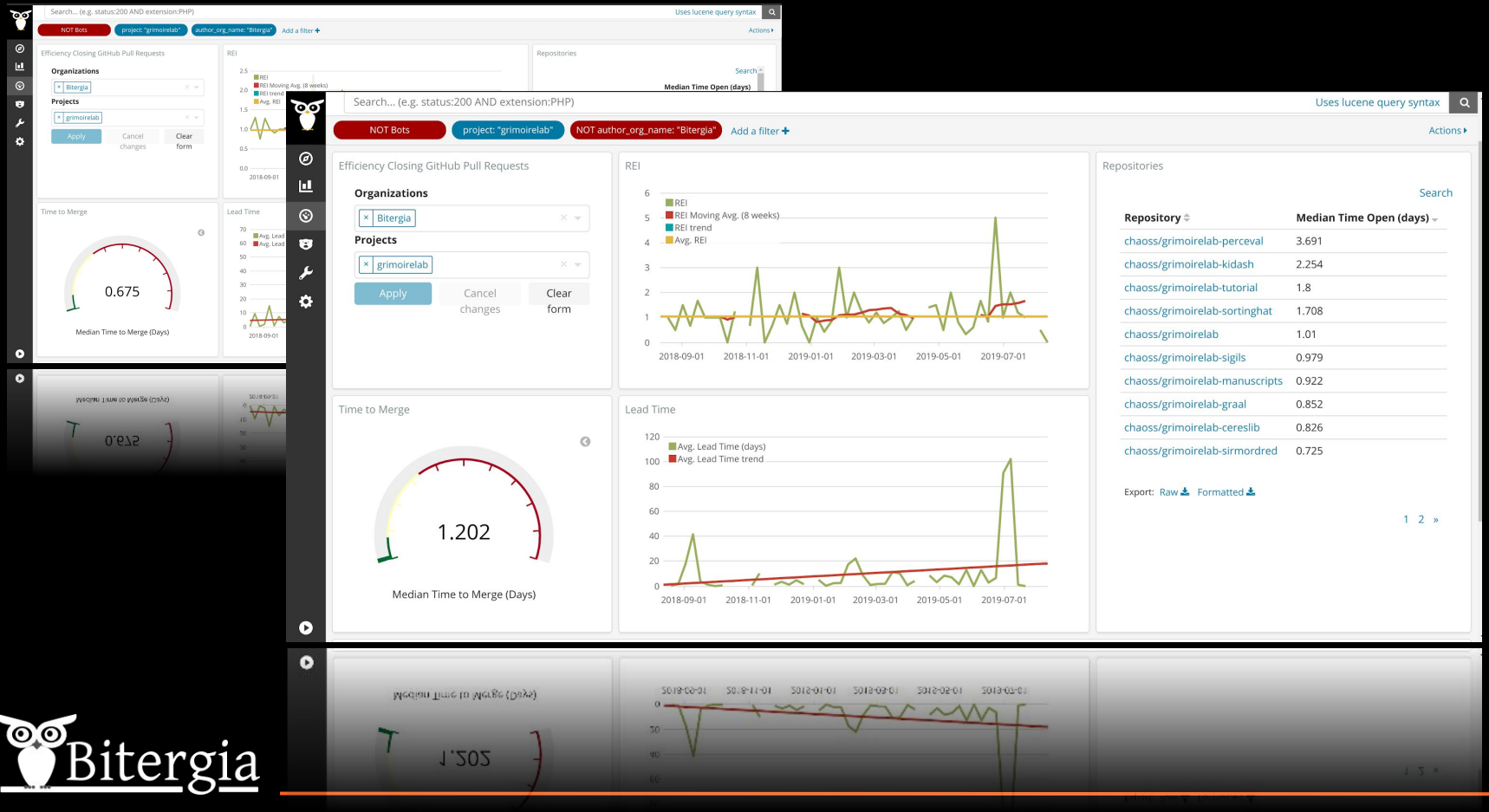
What is the responsiveness to CHAOSS community pull requests in GrimoireLab?

Lead time, time to 1st response, etc.



Bitergia

Lead time, time to 1st response, etc.

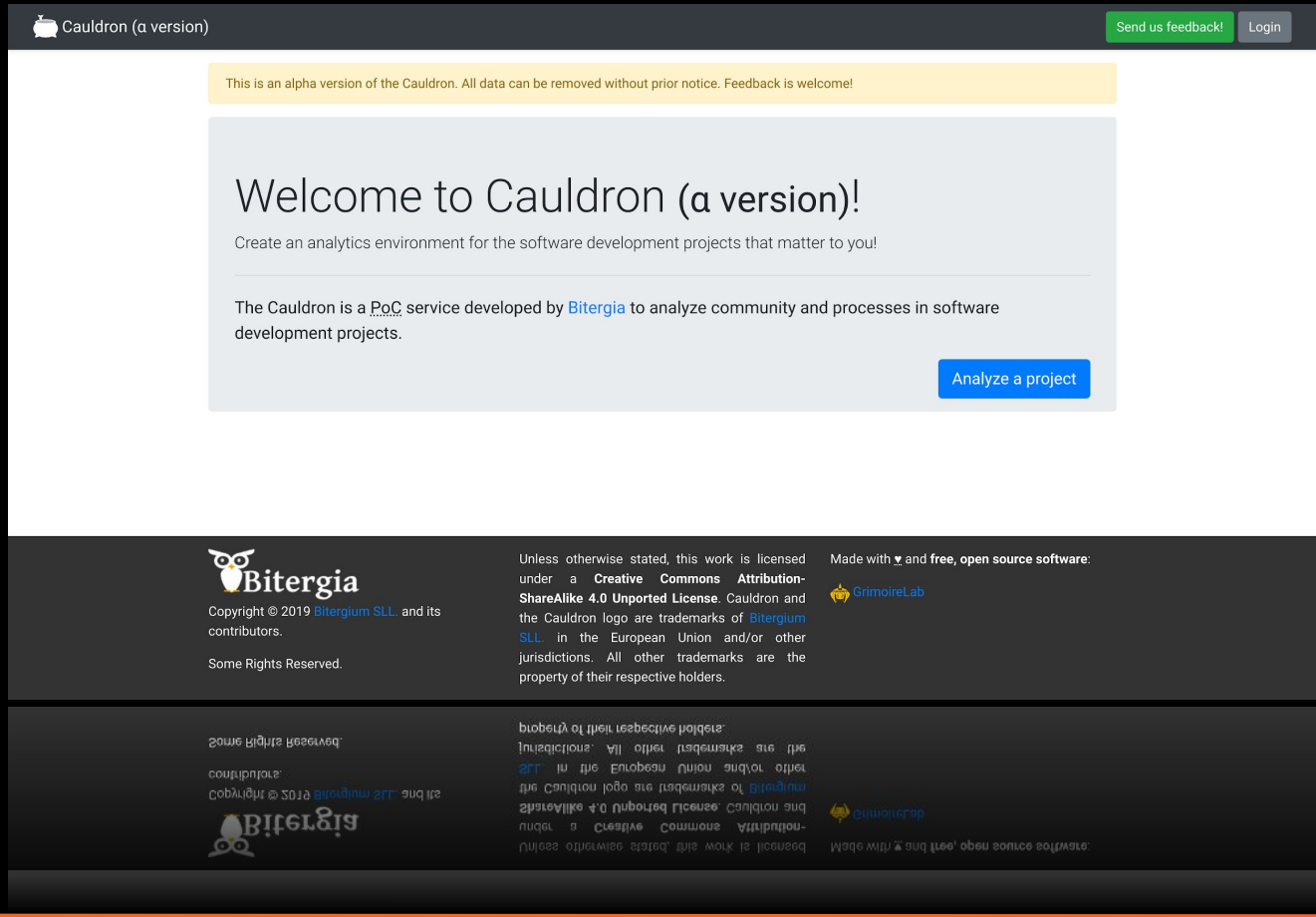


Bitergia

That is *almost* all ...

One more thing!

alpha.cauldron.io



Bitergia

Let’s go for questions!

Manrique López / **Santiago** Dueñas

CEO / **CTO** at **Bitergia**

jsmanrique@bitergia.com /
sduenas@bitergia.com

About us

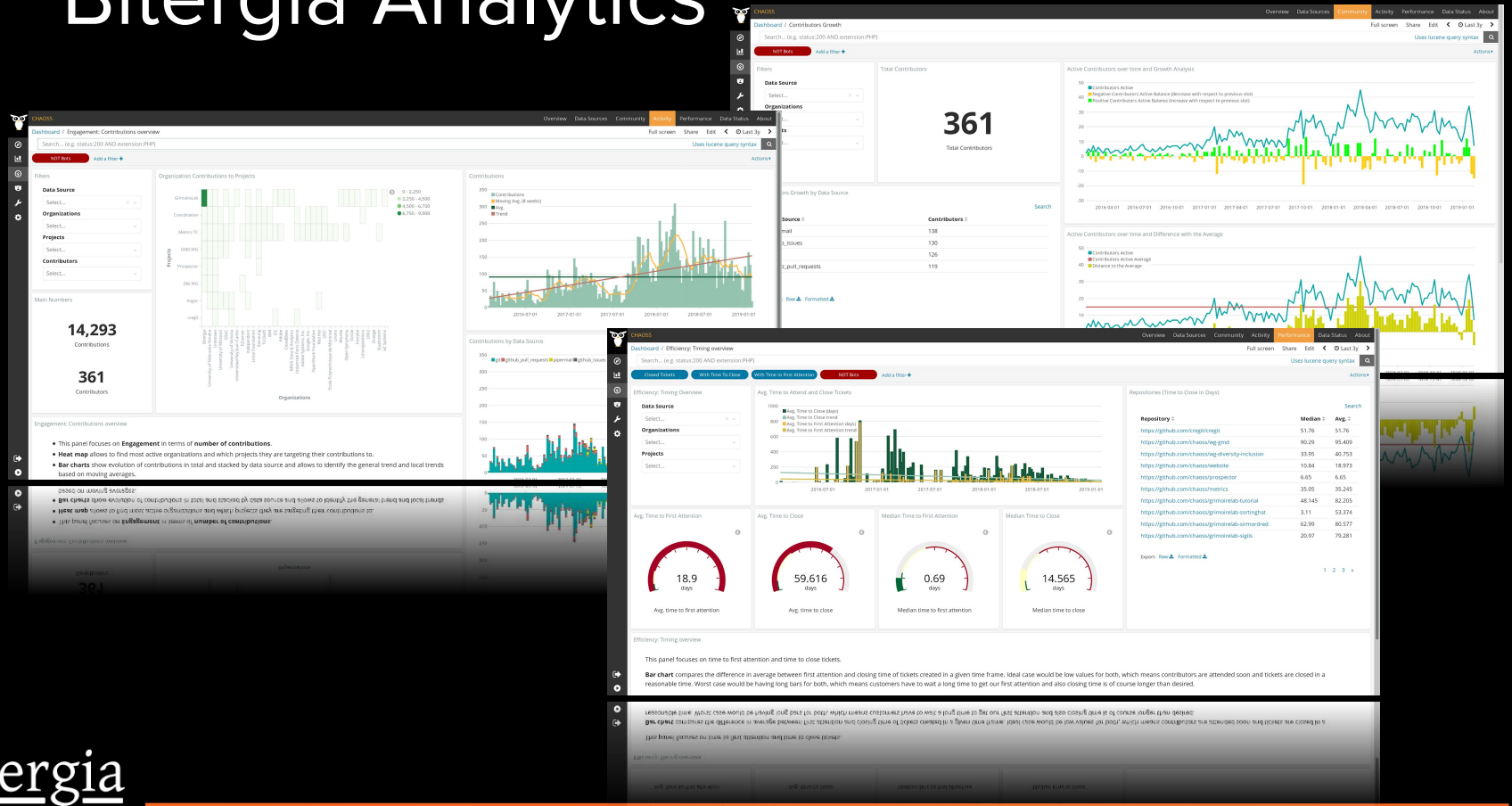
Bitergia

Software Development Analytics

For Your Peace of Mind

Bitergia helps companies and organizations with understanding and improving software development projects that matter to them

Bitergia Analytics



Bitergia

Community (who?)	• Who are the contributors to the analyzed projects? • Where are my developers? Where do they come from? • Who are my core, regular and casual developers? • What's the talent rotation and retention level?
Activity (what?)	• What is being done in the analyzed projects? • How many active projects do I contribute to? • What's developers engagement level? • What is being modified and what's left untouched for too long?
Performance (how?)	• How fast are projects analyzed performing? • How are we dealing with issues and merge requests? • Where are the bottlenecks? • How are we dealing with the backlog?

Community (who?)

- Who are the contributors to the analyzed projects?
- Where are my developers? Where do they come from?
- Who are my core, regular and casual developers?
- What's the talent rotation and retention level?

Activity (what)

- Wh[illegible] analyzed projects?
- [illegible] do I contribute to?
- W[illegible] ement level?
- [illegible]nd what's left untouched for too long?

Performance (how?)

- How fast are projects analyzed performing?
- How are we dealing with issues and merge requests?
- Where are the bottlenecks?
- How are we dealing with the backlog?

Bitergia

id law partners

How we do it

Bitergia Analytics Consultancy

Bitergia Analytics Platform

Strategy

Bitergia outlines organization strategy around software development to achieve organization's business goals.

Analysis

Bitergia defines the data sources, questions and associated metrics to measure that provide the insights about goals status.

Customization

Bitergia deploy and operates its analytics platform to gather the data needed to answer the questions and metrics defined.

Reporting

Bitergia provides consistent reporting mechanisms, including dashboards, reports, and even data APIs for custom integrations.

Open Source Software Foundations

Non-profit organizations managing open source projects.

Open Source Program Offices (OSPO)

Managing their relation with the open source projects they depend on.

InnerSource Program Offices (ISPO)

Adopting open source development practices internally.

Open Source Software Foundations

Non-profit organizations managing open source projects.

- Transparency level up
- Organizational diversity
- Members engagement
- Fair play among *coopetitors*
- Projects attraction and demographics
- Management board composition

Open Source Program Offices (OSPO)

Managing their relation with the open source projects they depend on.

- Company OSS ecosystem
- Talent acquisition and retention
- Company footprint in OSS
- Consistent reporting mechanism

InnerSource Program Offices (ISPO)

Adopting open source development practices internally.

- Developers engagement and talent retention
- Cross-Collaboration
- Onboarding mentoring
- Reuse and optimization


Strategy
Reporting
Bitergia Analytics Services
Analysis
Customization

Bitergia

“To measure is to know”

“If you can not measure it, you cannot improve it”

Lord Kelvin