

# Mining Software Repositories (MSR)

John Businge  
[\(john.businge@uantwerpen.be\)](mailto:(john.businge@uantwerpen.be))  
Henrique Rocha



# References

## The Road Ahead for Mining Software Repositories

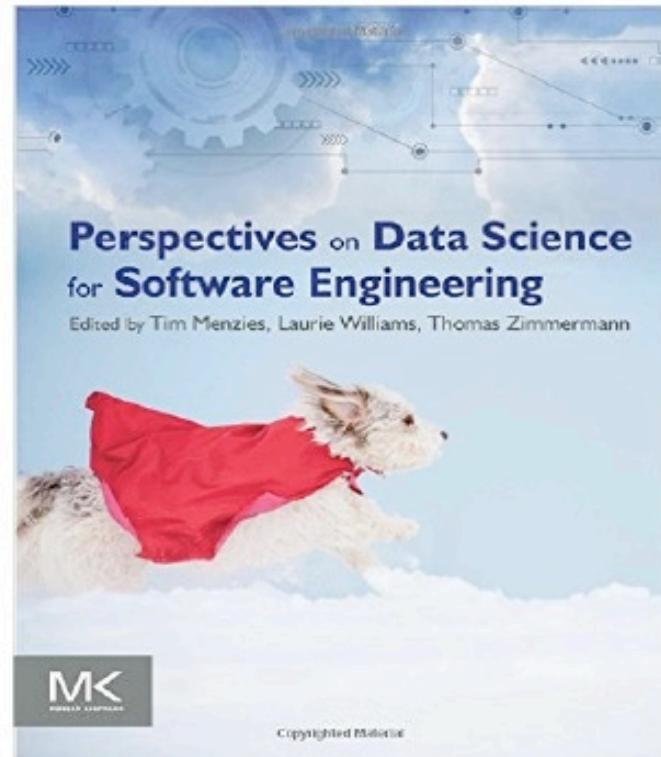
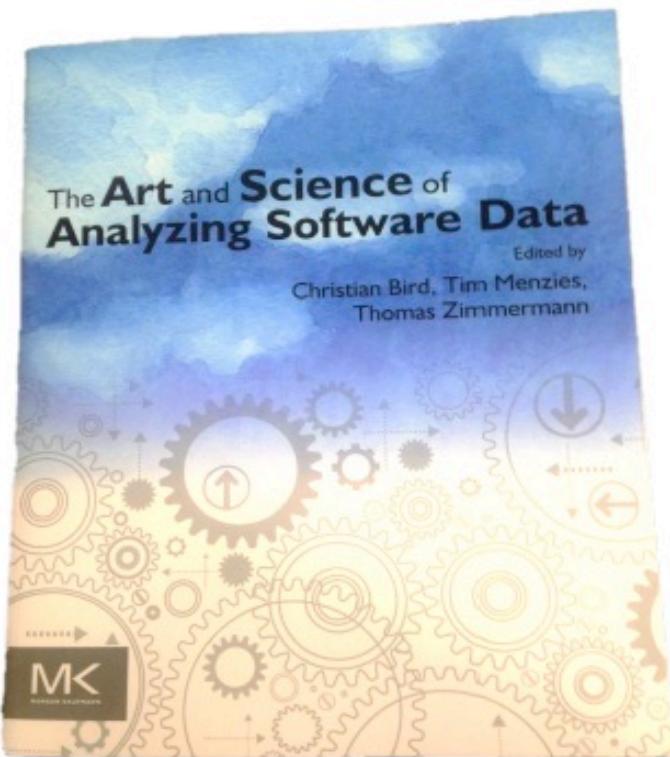
Ahmed E. Hassan  
Software Analysis and Intelligence Lab (SAIL)  
School of Computing, Queen's University, Canada  
[ahmed@cs.queensu.ca](mailto:ahmed@cs.queensu.ca)

## Software Intelligence: The Future of Mining Software Engineering Data

Ahmed E. Hassan  
School of Computing  
Queen's University  
Kingston, ON, Canada  
[ahmed@cs.queensu.ca](mailto:ahmed@cs.queensu.ca)

Tao Xie  
Department of Computer Science  
North Carolina State University  
Raleigh, NC, USA  
[xie@csc.ncsu.edu](mailto:xie@csc.ncsu.edu)

# More References



# Acknowledgement

**Ahmed E. Hassan**

Queen's University

[www.cs.queensu.ca/~ahmed](http://www.cs.queensu.ca/~ahmed)

[ahmed@cs.queensu.ca](mailto:ahmed@cs.queensu.ca)

**Tao Xie**

North Carolina State University

[www.csc.ncsu.edu/faculty/xie](http://www.csc.ncsu.edu/faculty/xie)

[xie@csc.ncsu.edu](mailto:xie@csc.ncsu.edu)

**Bram Adams**

Polytechnique Montréal, Canada

<http://mcis.polymtl.ca/index.html>

[lab.mcis@gmail.com](mailto:lab.mcis@gmail.com)

With updates by **John Businge** from University of Antwerp, Belgium

# Lecture Goals

---

- **Learn about:**
  - Classic and notable research and researchers in mining software engineering (SE) data
  - Data mining and data processing techniques and how to apply them to SE data
  - Risks in using SE data due to e.g., noise
- **After the lecture, you should be able to:**
  - Retrieve SE data
  - Prepare SE data for analysis
  - Mine interesting information from SE data

# Why mine SE data?

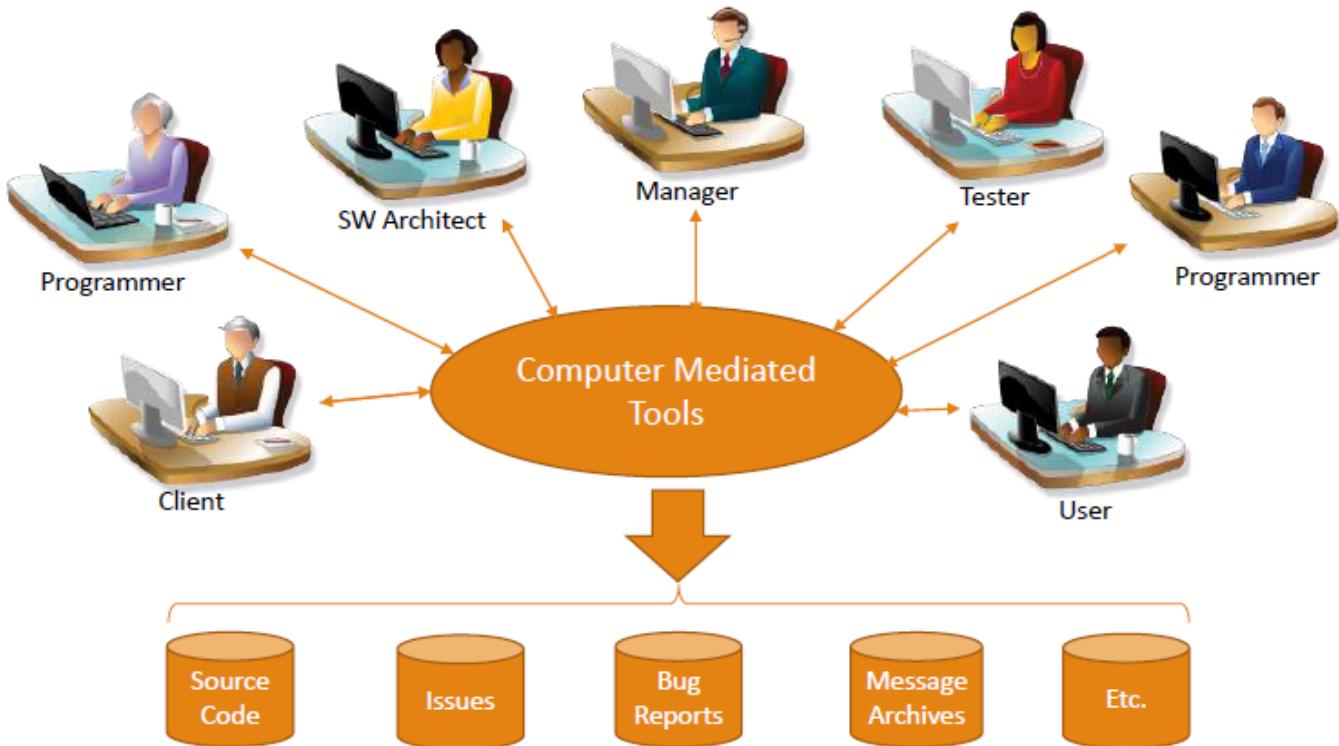
---

- **SE data can be used to:**

- Gain empirically-based understanding of software development
- Predict, plan, and understand various aspects of a project
- Support future development and project management activities



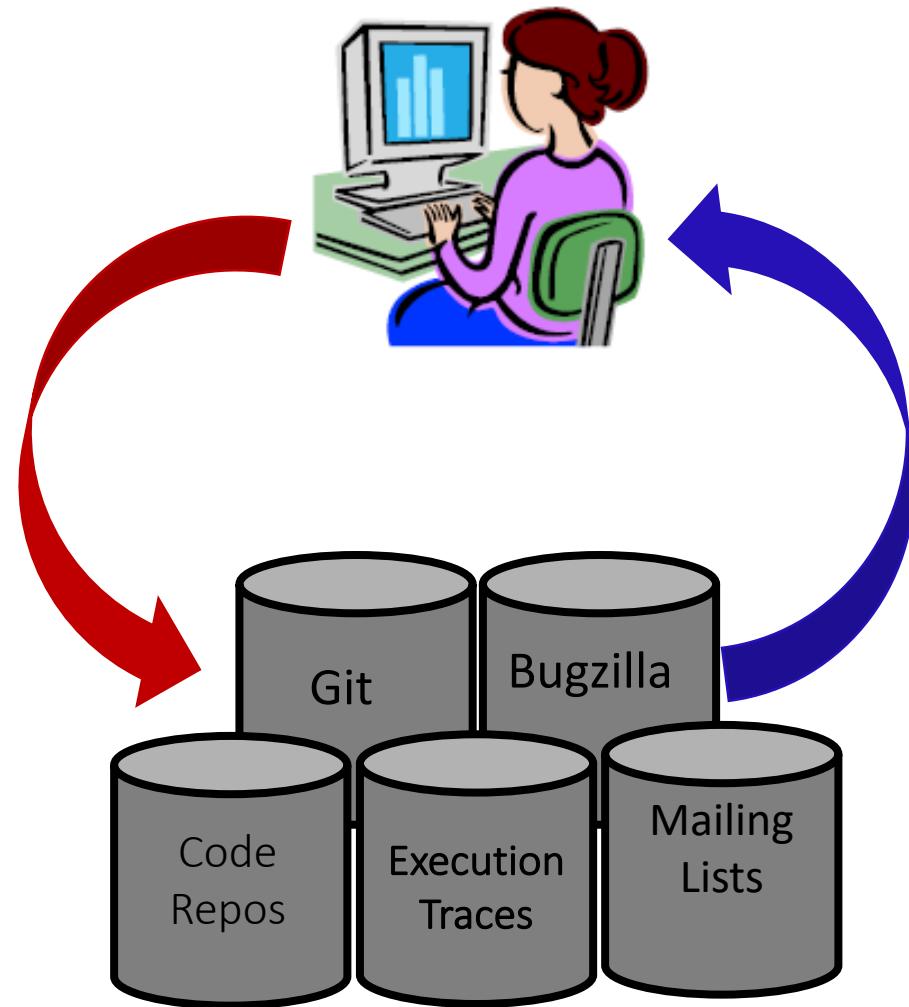
# How is SE Data generated?



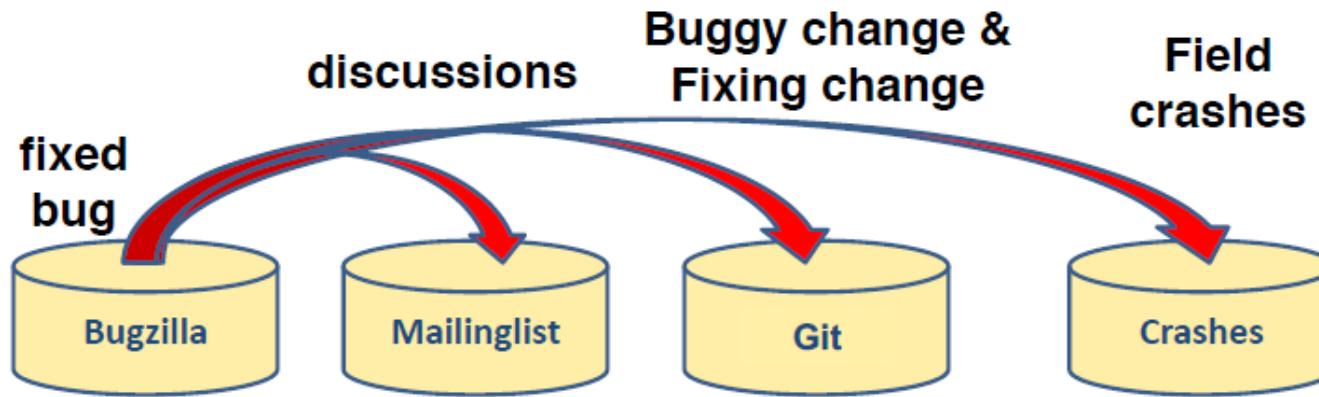
Current and historical artifacts and interactions are registered in software repositories

# What is MSR?

- Transforming static record keeping SE into active data
- Making SE data actionable by uncovering patterns and trends



# MSR researchers analyze and cross-link repositories



New bug report  
Estimate fix effort  
Suggest experts to fix the bugs!

# What can we learn from SE data?

---

- Let us look at a sample of notable findings from MSR studies.

# MSR studies – Bugs (1)

## Using imports to predict Bugs

- **MSR:** bugs and imports
- Trained models of No. of **bugs** vs No. of **imports**

[Schröter et al. 06]

### Results

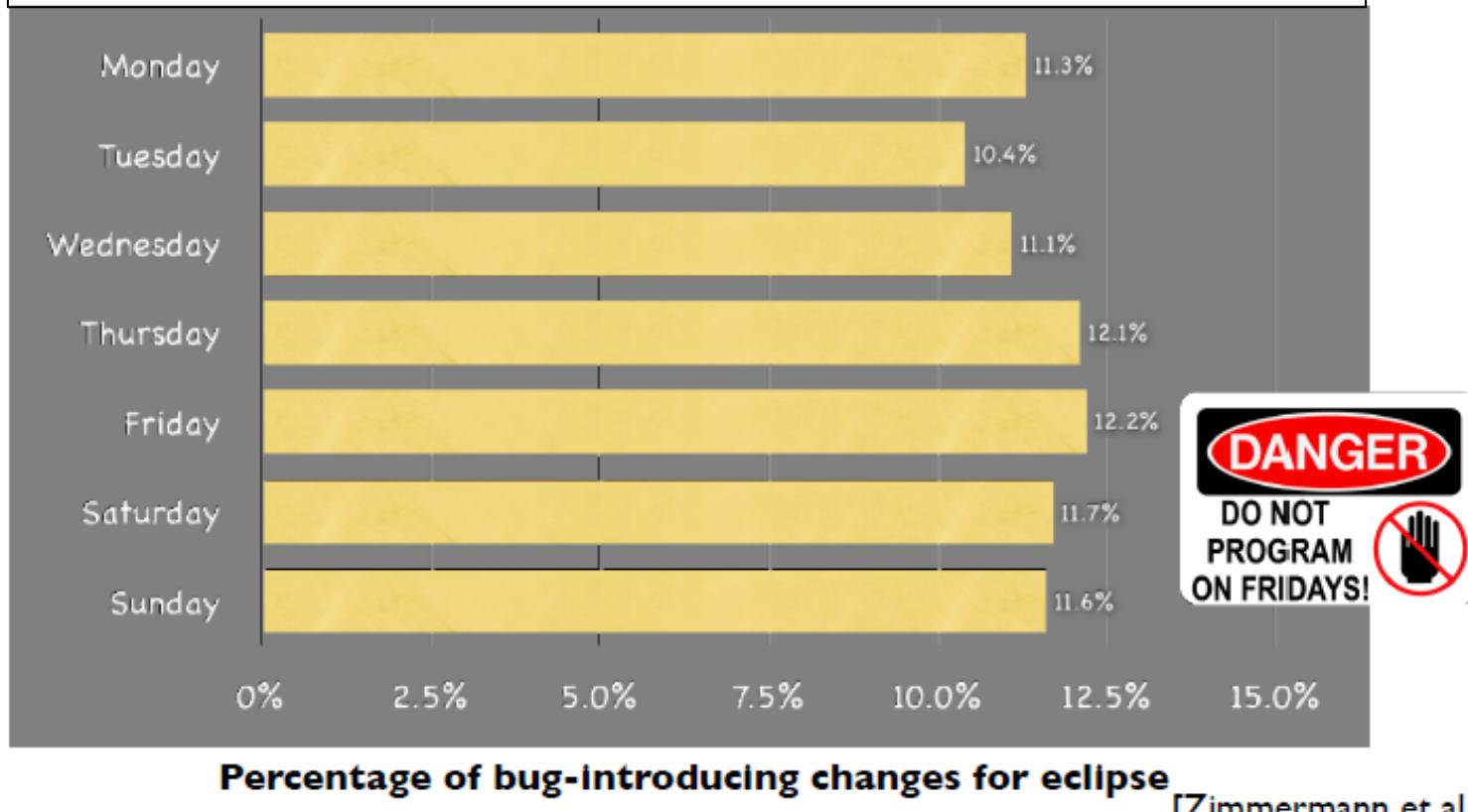
71% of files that import compiler packages,  
had to be fixed later on.

```
import org.eclipse.jdt.internal.compiler.lookup.*;  
import org.eclipse.jdt.internal.compiler.*;  
import org.eclipse.jdt.internal.compiler.ast.*;  
import org.eclipse.jdt.internal.compiler.util.*;  
...  
import org.eclipse.pde.core.*;  
import org.eclipse.jface.wizard.*;  
import org.eclipse.ui.*;
```

14% of all files that import ui packages, had  
to be fixed later on.

# MSR studies – Bugs (2)

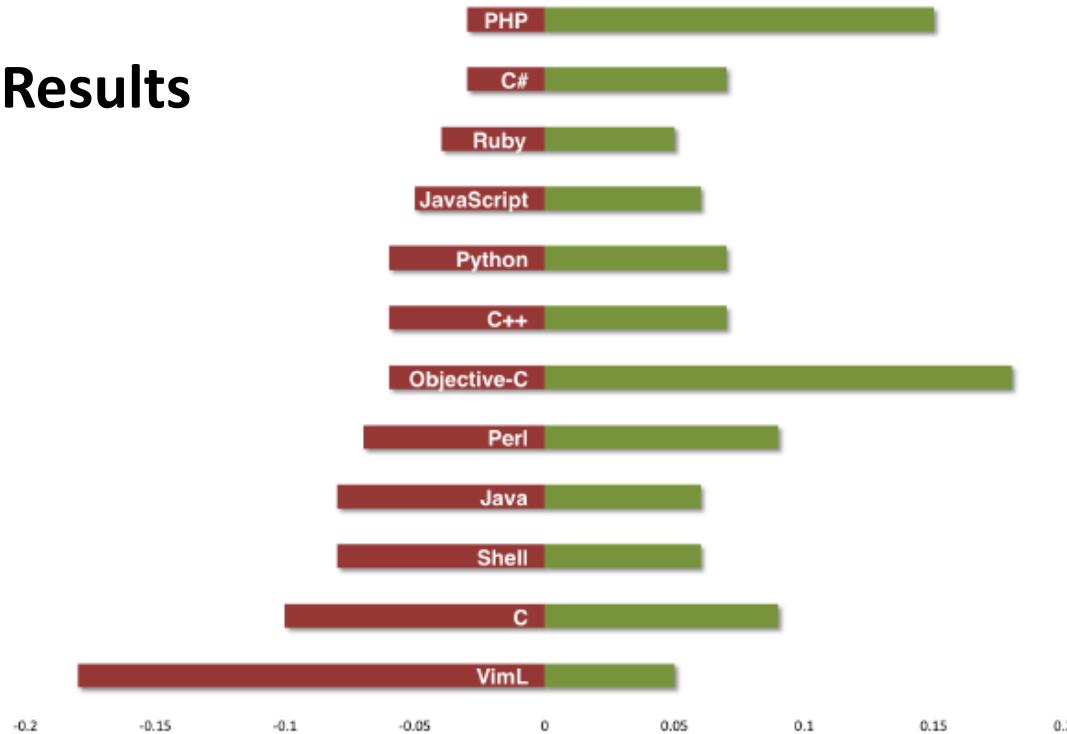
- **MSR:** CVS and Bugzilla (bug database) for the Eclipse framework
- **Results:** Fix-inducing changes showed a distinct pattern w.r.t days of the week.



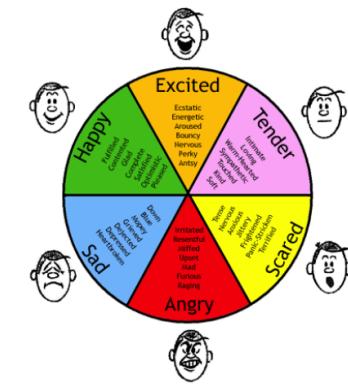
# MSR studies – Sentiment Analysis (1)

MSR: Mined commit logs, written in various programming languages and extracted phrases related to **Anger** and **Joy**

## Results



## Anger vs. Joy



## How they stack up?

- PHP, Object-C, and C# were net positive
- Java, Shell, and C were fairly even
- VimL is just bad news.

[Doll and Grigorik, 2012]

# MSR studies – Sentiment Analysis (2)

Ortu et al., MSR-2015

## Are Bullies more Productive? Empirical Study of Affectiveness vs. Issue Fixing Time

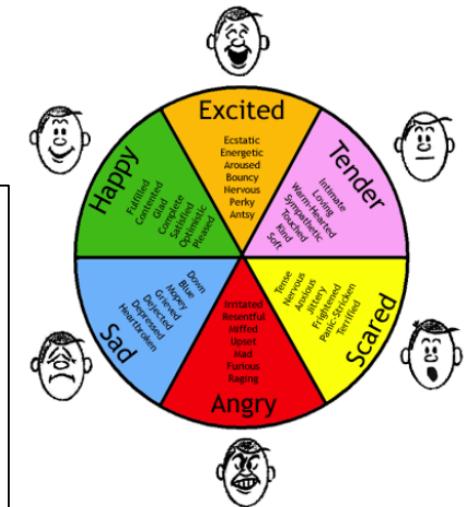
Marco Ortu\*, Bram Adams ‡, Giuseppe Destefanis†, Parastou Tourani ‡, Michele Marchesi \* Roberto Tonelli \*

\*DIEE, University of Cagliari, Italy, {marco.ortu,michele,roberto.tonelli}@diee.unica.it

†CRIM, Computer Research Institute of Montreal, Canada, {giuseppe.destefanis}@crim.ca

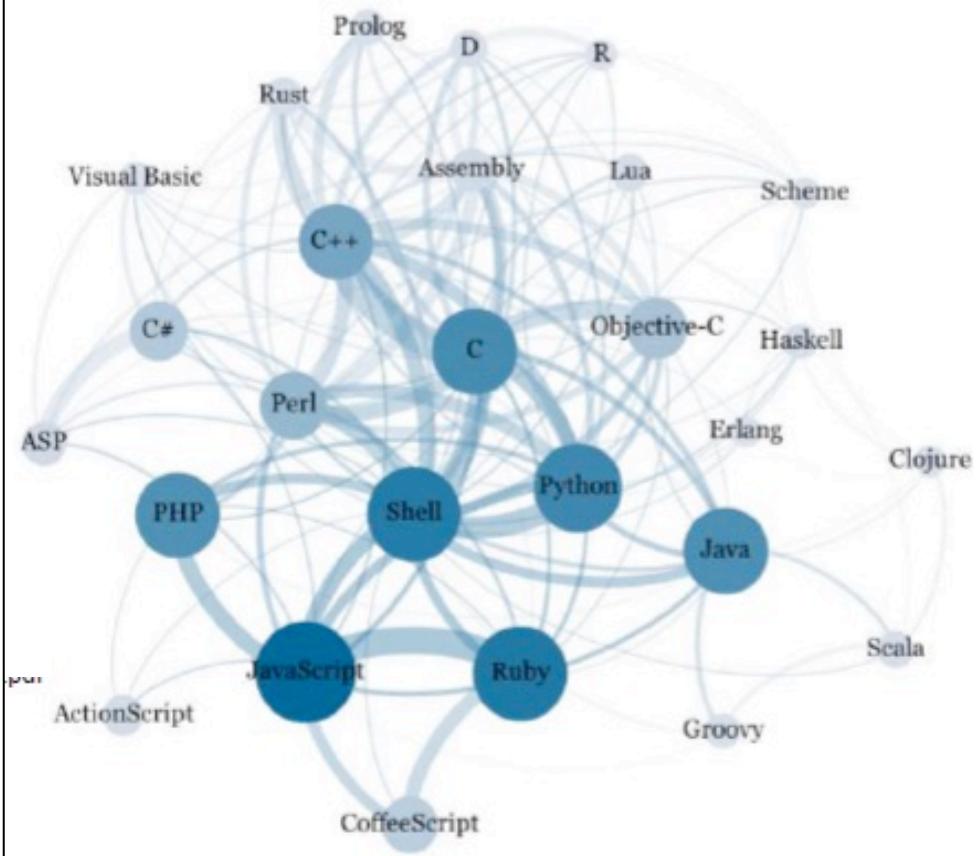
‡École Polytechnique de Montréal, Canada, {bram.adams,parastou.tourani}@polymtl.ca

- **MSR:** Mined more than **560K Jira comments** as well as the **issue fixing times**.
- **Results:** They found that the happier developers (expressing emotions such as **JOY** and **LOVE** in their comments) had shorter the issue fixing times.



# MSR studies – Programming languages

Visualising the GitHub programming language usage correlation dataset



- A **Ruby** programmers **is very likely to know JavaScript**, while a **Perl** programmer is not
- **Java** is a popular programming language but stands primarily alone

<https://github.com/mjwillson/ProgLangVisualise>

# MSR studies – Changes by programmers

## MSR to uncover programming language relations

The screenshot shows the Eclipse IDE interface with several windows:

- Java - Eclipse Platform**: The main window showing the menu bar (File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help) and a toolbar.
- Package Explorer**: Shows files like `CompareEditorContributor.java`, `CompareMessages.java`, `CompareNavigator.java`, and `ComparePreferencePage.java`.
- ComparePreferencePage.java**: An open editor window containing Java code. A red box labeled "A" highlights the line where a new preference is inserted into a field.
- Related Changes**: A view showing a list of changes. A red box labeled "B" highlights the entry for `initDefaults(IPreferenceStore store)`.

**A) The user inserts a new preference into the field fKeys[]**

```
public final OverlayPreferenceStore.OverlayKey[] fKeys = new OverlayPreferenceStore.OverlayKey[] {  
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.OPEN_STRUCTURE_COMFLICT, false),  
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.SHOW_PSEUDO_CONFLICTS, false),  
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.SHOW_ANCESTOR_PANE, false),  
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.SHOW_MORE_INFO, false),  
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.IGNORE_WHITESPACE, false),  
    new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.PREF_SAVE_ALL_EDIT, false)  
};  
  
new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.NEW_PREFERENCE, true);  
new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.STRING, AbstractTextEditor.class);  
new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, AbstractTextEditor.class);  
//new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, USER_SPLINES);  
new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, USER_SINGLE_LINE);  
//new OverlayPreferenceStore.OverlayKey(OverlayPreferenceStore.BOOLEAN, USER_RESOLVE_UI);  
};
```

**B) ROSE suggests locations for further changes, e.g. the function initDefaults()**

```
public static void initDefaults(IPreferenceStore store) {  
    store.setDefault(OPEN_STRUCTURE_COMFLICT, true);  
    store.setDefault(SYNCHRONIZE_SCROLLING, true);  
    store.setDefault(SHOW_PSEUDO_CONFLICTS, false);  
    store.setDefault(INITIALLY_SHOW_ANCESTOR_PANE, false);  
    store.setDefault(SHOW_MORE_INFO, false);  
    store.setDefault(IGNORE_WHITESPACE, false);  
    store.setDefault(PREF_SAVE_ALL_EDITORS, false);  
    //store.setDefault(USER_SPLINES, false);  
    store.setDefault(USER_SINGLE_LINE, true);  
}
```

[Zimmermann et al., 2005]

After the programmer has made some changes to the source (above), ROSE suggests locations where further changes were made

### ROSE tool guides the programmer by:

- Suggesting and predicting likely changes
- Preventing errors due to incomplete changes

# Where can we mine SE Data?

---



Sept 2020:

100 Millinon repositories  
56 Million Users



April 2019

28 Millinon repositories  
10 Million Users



Travis CI

Libraries.io

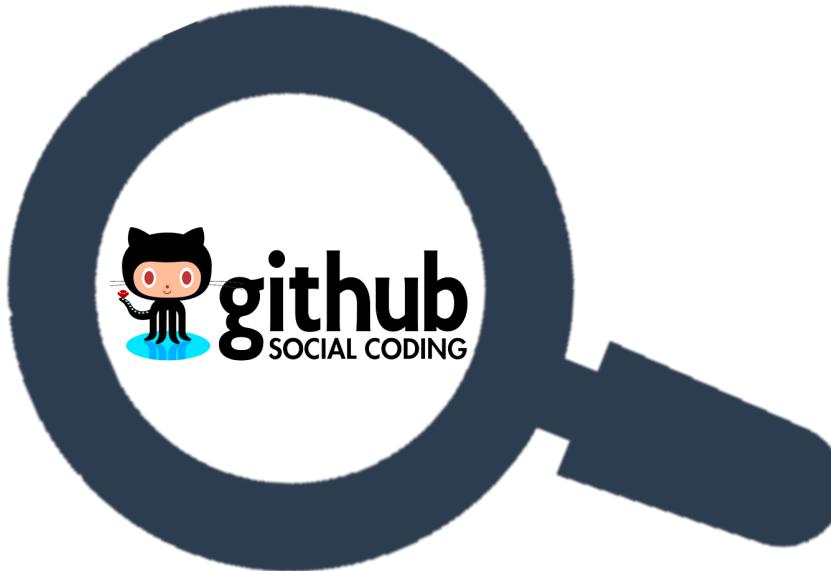


# How can we mine SE Data?

---



All the mentioned platforms and many more have got REST APIs



# How can we mine GitHub Data?



Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.



Easiest is to clone & own a repo from GitHub. The cloned copy will have most of the data from the archive. However, branches can be pain to analyse!



# How can we mine GitHub Data?

The screenshot shows a GitHub user profile for 'bramadams'. At the top, there's a red bar. Below it is the GitHub header with a search bar, navigation links (Explore, Gist, Blog, Help), and user account options. The main content area features a 'GitHub Bootcamp' card with four numbered steps: 1. Set up Git, 2. Create repositories, 3. Fork repositories, and 4. Work together. Each step has an illustration and a brief description. Below the bootcamp card is a large yellow box containing the text: 'Access to thousands of Git-based projects directly from GitHub'. To the right of this yellow box are several notifications: 'You've been mentioned in a broadcast', 'Better Word Highlighting In Diffs', and 'View 70 new broadcasts'. At the bottom, there's a section titled 'Repositories you contribute to' with two listed repositories: 'inukshuk/Jekyll-scholar' and 'smcintosh/moosetracks'.

Search GitHub

Explore Gist Blog Help

bramadams + ⌂ ⚙ ⌂

bramadams News Feed Pull Requests Issues

**GitHub Bootcamp**

- 1 Set up Git  
A quick guide to help you get started with Git.
- 2 Create repositories  
Repositories are where you'll work and collaborate on projects.
- 3 Fork repositories  
Forking creates a new, unique project from an existing one.
- 4 Work together  
Send pull requests, follow friends. Star and watch projects.

You've been mentioned in a broadcast

Better Word Highlighting In Diffs

Commits, compare views, and pull requests now highlight individual changed words.

View 70 new broadcasts

Repositories you contribute to

- inukshuk/Jekyll-scholar 152 ★
- smcintosh/moosetracks 1 ★

# How can we mine GitHub Data?

The screenshot shows the GitHub REST API documentation page. On the left, there is a dark sidebar with the GitHub logo and the text "GitHub Docs". Below the logo is a list of navigation links: Gitignore, Interactions, Issues, Licenses, Markdown, Meta, Migrations, Organizations, Packages, Projects, Pulls, Rate limit, Reactions, Repositories, SCIM, Search, Secret scanning, Teams, Users, Permissions required for GitHub Apps, and GUIDES. At the bottom of the sidebar, it says "Getting started with the REST API". On the right, the main content area has a header "REST API" and a large section titled "GitHub REST API" with a print icon. Below this, there is a paragraph: "You can use the GitHub REST API to create calls to get the data you need to integrate with GitHub." Underneath this paragraph are three sections: "REST API overview", "Reference", and "Guides". Each section has a brief description and a link to the full documentation.

REST API

## GitHub REST API

You can use the GitHub REST API to create calls to get the data you need to integrate with GitHub.

[REST API overview](#)

Learn about resources, libraries, previews and troubleshooting for GitHub's REST API.

[Reference](#)

View reference documentation to learn about the resources available in the GitHub REST API.

[Guides](#)

Learn about getting started with the REST API, authentication, and how to use the REST API for a variety of tasks.

# How can we mine GitHub Data?

## List repository contributors

Lists contributors to the specified repository and sorts them by the number of commits per contributor in descending order. This endpoint may return information that is a few hours old because the GitHub REST API v3 caches contributor data to improve performance.

GitHub identifies contributors by author email address. This endpoint groups contribution counts by GitHub user, which includes all associated email addresses. To improve performance, only the first 500 author email addresses in the repository link to GitHub users. The rest will appear as anonymous contributors without associated GitHub user information.

GET /repos/{owner}/{repo}/contributors

### Parameters

Name	Type	In	Description
accept	string	header	Setting to <code>application/vnd.github.v3+json</code> is recommended.
owner	string	path	
repo	string	path	
anon	string	query	Set to <code>1</code> or <code>true</code> to include anonymous contributors in results.
per_page	integer	query	Results per page (max 100).
page	integer	query	Page number of the results to fetch.

List organization repositories

Create an organization repository

Get a repository

Update a repository

Delete a repository

Enable automated security fixes

Disable automated security fixes

List repository contributors

Create a repository dispatch event

List repository languages

List repository tags

List repository teams

Get all repository topics

Replace all repository topics

Transfer a repository

Check if vulnerability alerts are enabled for a repository

Enable vulnerability alerts

Disable vulnerability alerts

Create a repository using a template

List public repositories

List repositories for the authenticated user

Create a repository for the authenticated user

List repositories for a user

### Branches

List branches

Get a branch

Get branch protection

Update branch protection

# How can we mine GitHub Data

The screenshot shows the GitHub Docs Reference page. The left sidebar contains a navigation menu with links like All products, REST API, OVERVIEW, and a large REFERENCE section containing links for Actions, Activity, Apps, Billing, Checks, Codes of conduct, Code scanning, Emojis, GitHub Enterprise administration, Gists, Git database, Gitignore, Interactions, Issues, Licenses, Markdown, Meta, Migrations, Organizations, Packages, Projects, and Pulls. The main content area has a heading Reference and a sub-section Table of Contents with a list of the same items. A red callout box on the right side contains the text: There are rate-limit issues! You are only allowed only a limited number of GitHub requests per hour.

GitHub Docs

All products

REST API

OVERVIEW

REFERENCE

Actions

Activity

Apps

Billing

Checks

Codes of conduct

Code scanning

Emojis

GitHub Enterprise administration

Gists

Git database

Gitignore

Interactions

Issues

Licenses

Markdown

Meta

Migrations

Organizations

Packages

Projects

Pulls

## Reference

View reference documentation to learn about the resources available in the GitHub REST API.

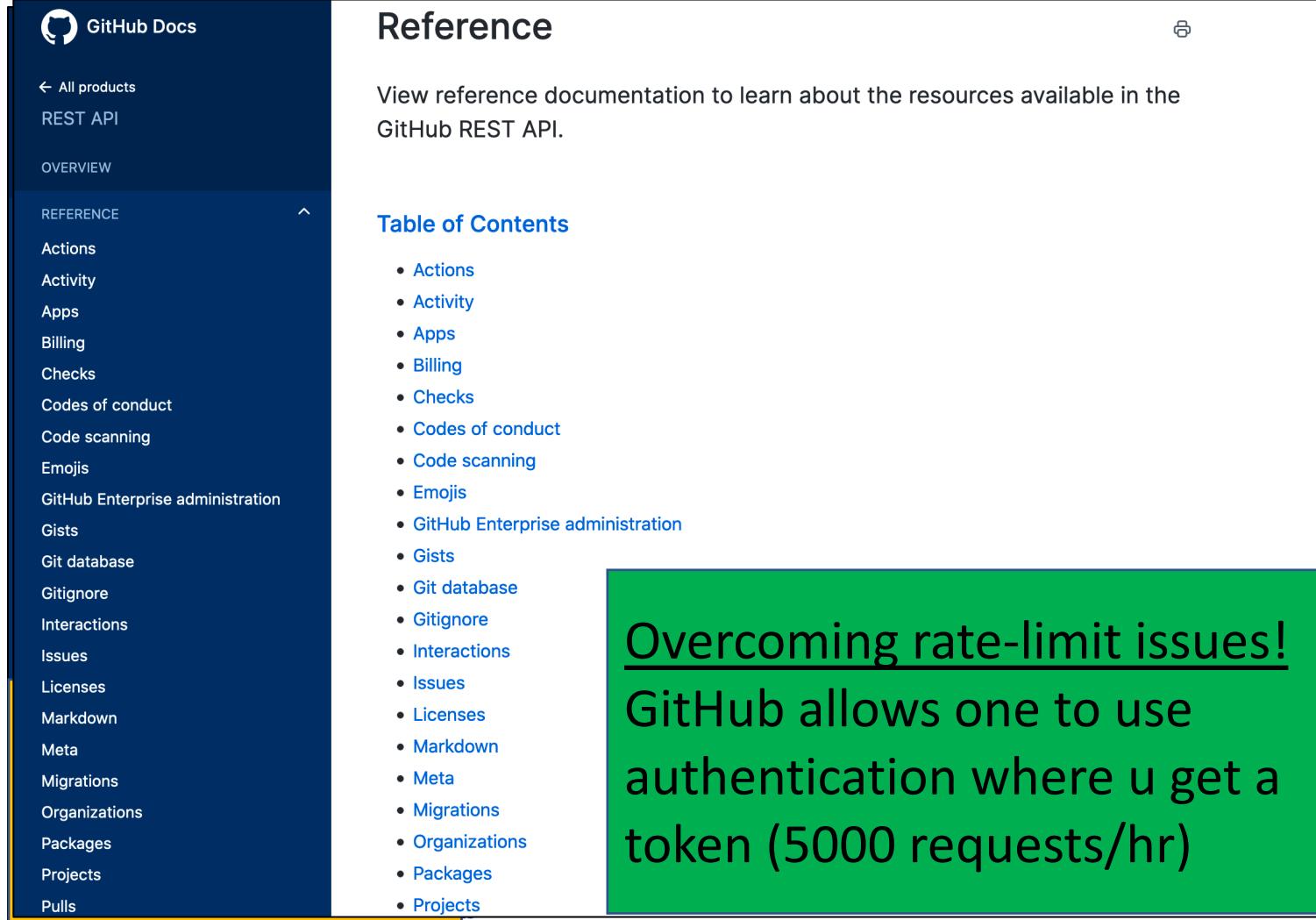
### Table of Contents

- Actions
- Activity
- Apps
- Billing
- Checks
- Codes of conduct
- Code scanning
- Emojis
- GitHub Enterprise administration
- Gists
- Git database
- Gitignore
- Interactions
- Issues
- Licenses
- Markdown
- Meta
- Migrations
- Organizations
- Packages
- Projects

There are rate-limit issues!

You are only allowed only a limited number of GitHub requests per hour

# How can we mine GitHub Data



The screenshot shows the GitHub Docs interface for the REST API Reference. The left sidebar is dark blue with white text, listing various API endpoints like Actions, Activity, Apps, Billing, Checks, etc. The main content area has a white background with a title 'Reference' and a sub-section 'Table of Contents' which lists all the same endpoints. A green callout box on the right side contains the text: 'Overcoming rate-limit issues! GitHub allows one to use authentication where u get a token (5000 requests/hr)'. There is also a small red bar at the top of the slide.

## Reference

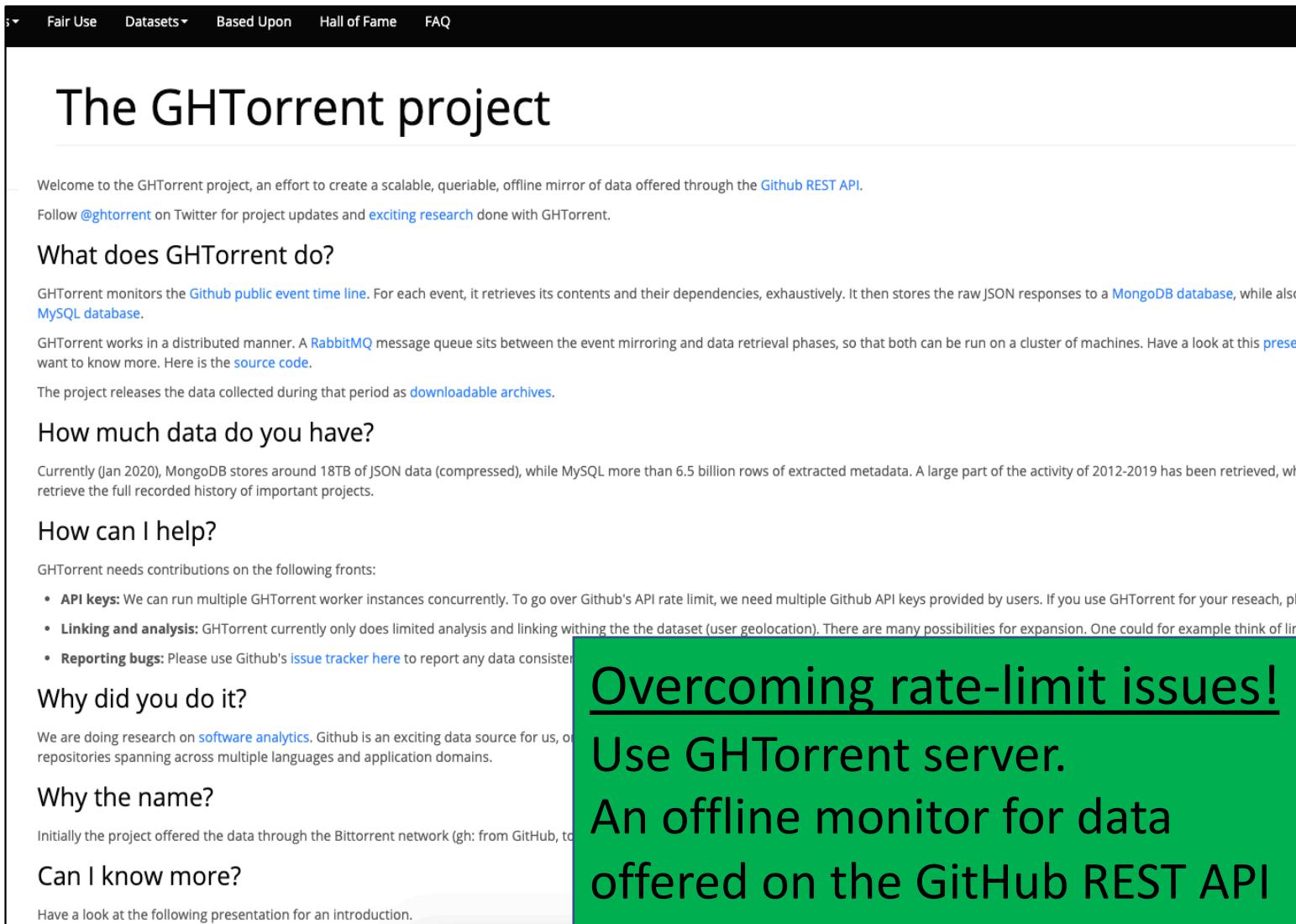
View reference documentation to learn about the resources available in the GitHub REST API.

### Table of Contents

- Actions
- Activity
- Apps
- Billing
- Checks
- Codes of conduct
- Code scanning
- Emojis
- GitHub Enterprise administration
- Gists
- Git database
- Gitignore
- Interactions
- Issues
- Licenses
- Markdown
- Meta
- Migrations
- Organizations
- Packages
- Projects

Overcoming rate-limit issues!  
GitHub allows one to use authentication where u get a token (5000 requests/hr)

# How can we mine GitHub Data



The screenshot shows a website for the GHTorrent project. At the top, there's a navigation bar with links for Fair Use, Datasets, Based Upon, Hall of Fame, and FAQ. Below the navigation, the main title "The GHTorrent project" is displayed in a large, bold font. A sub-section titled "What does GHTorrent do?" follows, containing text about monitoring GitHub events and storing data in MongoDB and MySQL databases. Another section, "How much data do you have?", provides information about the size of the collected data. A "How can I help?" section lists contributions needed, such as API keys and linking analysis. A "Why did you do it?" section discusses research on software analytics. A "Why the name?" section explains the origin of the project name. A "Can I know more?" section offers a link to an introduction presentation. On the right side of the page, there's a green sidebar with white text containing a summary of the project's purpose: "Overcoming rate-limit issues! Use GHTorrent server. An offline monitor for data offered on the GitHub REST API".

Fair Use   Datasets▼   Based Upon   Hall of Fame   FAQ

## The GHTorrent project

Welcome to the GHTorrent project, an effort to create a scalable, queriable, offline mirror of data offered through the [Github REST API](#). Follow [@ghtorrent](#) on Twitter for project updates and [exciting research](#) done with GHTorrent.

### What does GHTorrent do?

GHTorrent monitors the [Github public event time line](#). For each event, it retrieves its contents and their dependencies, exhaustively. It then stores the raw JSON responses to a [MongoDB database](#), while also [MySQL database](#).

GHTorrent works in a distributed manner. A [RabbitMQ](#) message queue sits between the event mirroring and data retrieval phases, so that both can be run on a cluster of machines. Have a look at this [presentation](#) if you want to know more. Here is the [source code](#).

The project releases the data collected during that period as [downloadable archives](#).

### How much data do you have?

Currently (Jan 2020), MongoDB stores around 18TB of JSON data (compressed), while MySQL more than 6.5 billion rows of extracted metadata. A large part of the activity of 2012-2019 has been retrieved, which allows us to retrieve the full recorded history of important projects.

### How can I help?

GHTorrent needs contributions on the following fronts:

- **API keys:** We can run multiple GHTorrent worker instances concurrently. To go over Github's API rate limit, we need multiple Github API keys provided by users. If you use GHTorrent for your research, please consider sharing your keys.
- **Linking and analysis:** GHTorrent currently only does limited analysis and linking within the dataset (user geolocation). There are many possibilities for expansion. One could for example think of linking to external datasets.
- **Reporting bugs:** Please use Github's [issue tracker here](#) to report any data consistency issues or other bugs.

### Why did you do it?

We are doing research on [software analytics](#). Github is an exciting data source for us, offering a wide variety of data across many different repositories spanning across multiple languages and application domains.

### Why the name?

Initially the project offered the data through the BitTorrent network (gh: from GitHub, to Torrent).

### Can I know more?

Have a look at the following presentation for an introduction.

## Overcoming rate-limit issues!

## Use GHTorrent server.

## An offline monitor for data offered on the GitHub REST API

# How can we mine SE Data – Part II

## Package managers - Libraries.io

 <b>Go</b> 1.82M Packages	 <b>npm</b> 1.32M Packages	 <b>Packagist</b> 321K Packages	 <b>PyPI</b> 242K Packages
 <b>NuGet</b> 201K Packages	 <b>Maven</b> 185K Packages	 <b>Rubygems</b> 164K Packages	 <b>Bower</b> 69.7K Packages
 <b>CocoaPods</b> 69.4K Packages	 <b>WordPress</b> 66.3K Packages	 <b>Cargo</b> 38.2K Packages	 <b>CPAN</b> 37.7K Packages
 <b>Clojars</b> 24.3K Packages	 <b>CRAN</b> 17K Packages	 <b>Hackage</b> 14.7K Packages	 <b>Meteor</b> 13.4K Packages
 <b>Atom</b> 12.9K Packages	 <b>Pub</b> 11.1K Packages	 <b>Hex</b> 9.7K Packages	 <b>PlatformIO</b> 6.86K Packages
 <b>Puppet</b> 6.51K Packages	 <b>Emacs</b> 4.9K Packages	 <b>Homebrew</b> 4.7K Packages	 <b>SwiftPM</b> 4.21K Packages
 <b>Carthage</b> 3.97K Packages	 <b>Julia</b> 3.05K Packages	 <b>Sublime</b> 2.01K Packages	 <b>conda</b> 1.97K Packages
 <b>Dub</b> 1.94K Packages	 <b>Racket</b> 1.72K Packages	 <b>Elm</b> 1.51K Packages	 <b>Haxelib</b> 1.45K Packages
 <b>Nimble</b> 1.26K Packages	 <b>Jam</b> 772 Packages	 <b>Alcatraz</b> 464 Packages	 <b>PureScript</b> 389 Packages
 <b>Inqlude</b> 224 Packages	 <b>Shards</b> 33 Packages		

# How can we mine SE Data from package managers?

react DT

17.0.1 • Public • Published 4 months ago

[Readme](#) [Explore BETA](#) [2 Dependencies](#) [68,550 Dependents](#) [340 Versions](#)

## react

React is a JavaScript library for creating user interfaces.

The `react` package contains only the functionality necessary to define React components. It is typically used together with a React renderer like `react-dom` for the web, or `react-native` for the native environments.

**Note:** by default, React will be in development mode. The development version includes extra warnings about common mistakes, whereas the production version includes extra performance optimizations and strips all error messages. Don't forget to use the `production build` when deploying your application.

### Example Usage

```
var React = require('react');
```

### Keywords

Install  
`> npm i react`

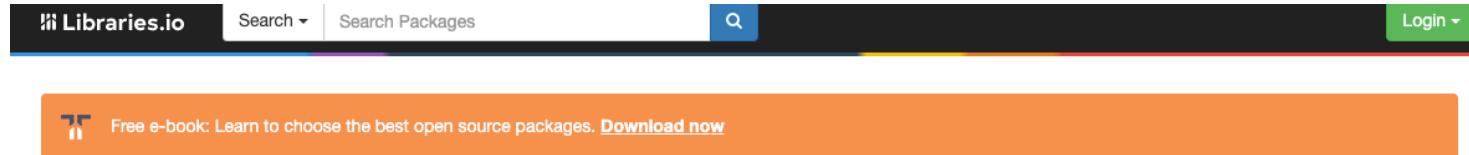
Weekly Downloads  
10,500,838 

Version	License
17.0.1	MIT
Unpacked Size	Total Files
298 kB	18
Issues	Pull Requests
565	168

Homepage  
[reactjs.org/](https://reactjs.org/)

Repository  
[github.com/facebook/react](https://github.com/facebook/react)

# How can we mine SE Data from package managers?



## API Docs

### Authentication

All API requests must include `api_key` parameter, get your api key from your [account page](#)

### Rate limit

All requests are subject to a 60/request/minute rate limit based on your `api_key`, any further requests within that timeframe will result in a `429` response.

### Pagination

All requests that return multiple results can be paginated using the `'page'` and `'per_page'` query parameters.

- `page` (default is '1')
- `per_page` (default is '30', max is '100')

### Platforms

Get a list of supported package managers.

`GET https://libraries.io/api/platforms?api_key=YOUR_API_KEY`

Example: `https://libraries.io/api/platforms?api_key=YOUR_API_KEY`

```
[  
 {  
   "name": "Go",  
   "project_count": 1818642,  
   "homepage": "http://go-search.org/",  
   "color": "#375eab",  
   "default_language": null  
 }
```

### API Methods

- Authentication
- Rate limit
- Pagination
- Platforms
- Project
- Project Dependencies
- Project Dependents
- Project Dependent Repositories
- Project Contributors
- Project SourceRank
- Project Usage
- Project Search
- Repository
- Repository Dependencies
- Repository Projects
- User
- User Repositories
- User Projects
- User Package Contributions
- User Repository Contributions
- User Dependencies
- User Subscriptions
- Subscribe to a project
- Check if subscribed to a project
- Update a subscription
- Unsubscribe from a project
- Wrappers

For any questions, feature requests or bug reports  
email [support@libraries.io](mailto:support@libraries.io) or [open an issue](#).

Let's get our hands dirty with the  
labs exercise

