Mining Software Repositories

John Businge

Henrique Rocha

Universiteit Antwerpen

References

The Road Ahead for Mining Software Repositories

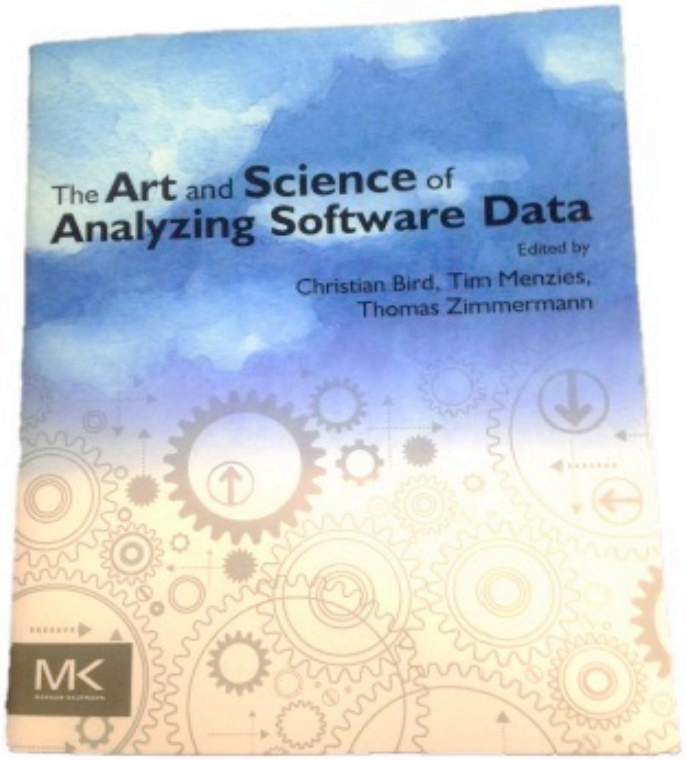
Ahmed E. Hassan
Software Analysis and Intelligence Lab (SAIL)
School of Computing, Queen's University, Canada
ahmed@cs.queensu.ca

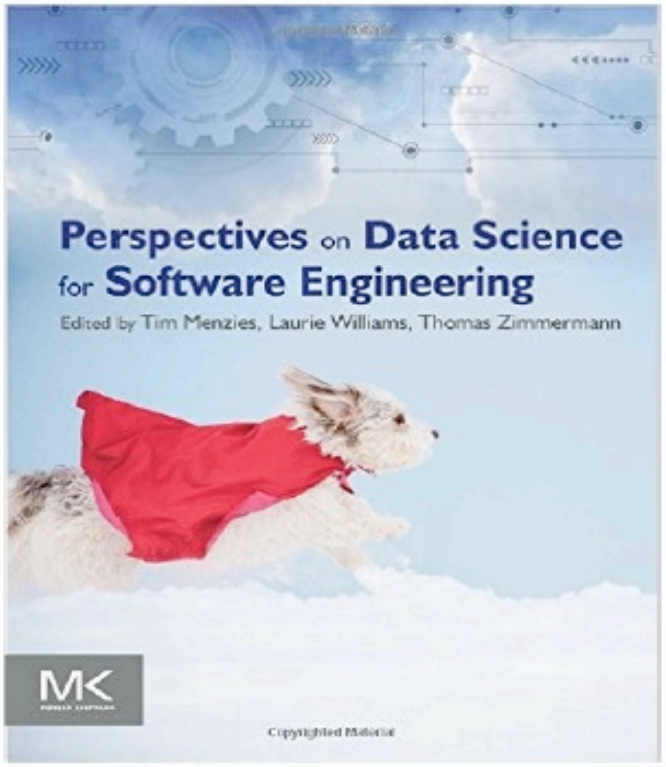
Software Intelligence:
The Future of Mining Software Engineering Data

Ahmed E. Hassan
School of Computing
Queen's University
Kingston, ON, Canada
ahmed@cs.queensu.ca

Tao Xie
Department of Computer Science
North Carolina State University
Raleigh, NC, USA
xie@csc.ncsu.edu

More References





Acknowledgement

Ahmed E. Hassan
Queen's University
www.cs.queensu.ca/~ahmed
ahmed@cs.queensu.ca

Tao Xie
North Carolina State University
www.csc.ncsu.edu/faculty/xie
xie@csc.ncsu.edu

Bram Adams
Polytechnique Montréal, Canada
http://mcis.polymtl.ca/index.html
lab.mcis@gmail.com

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

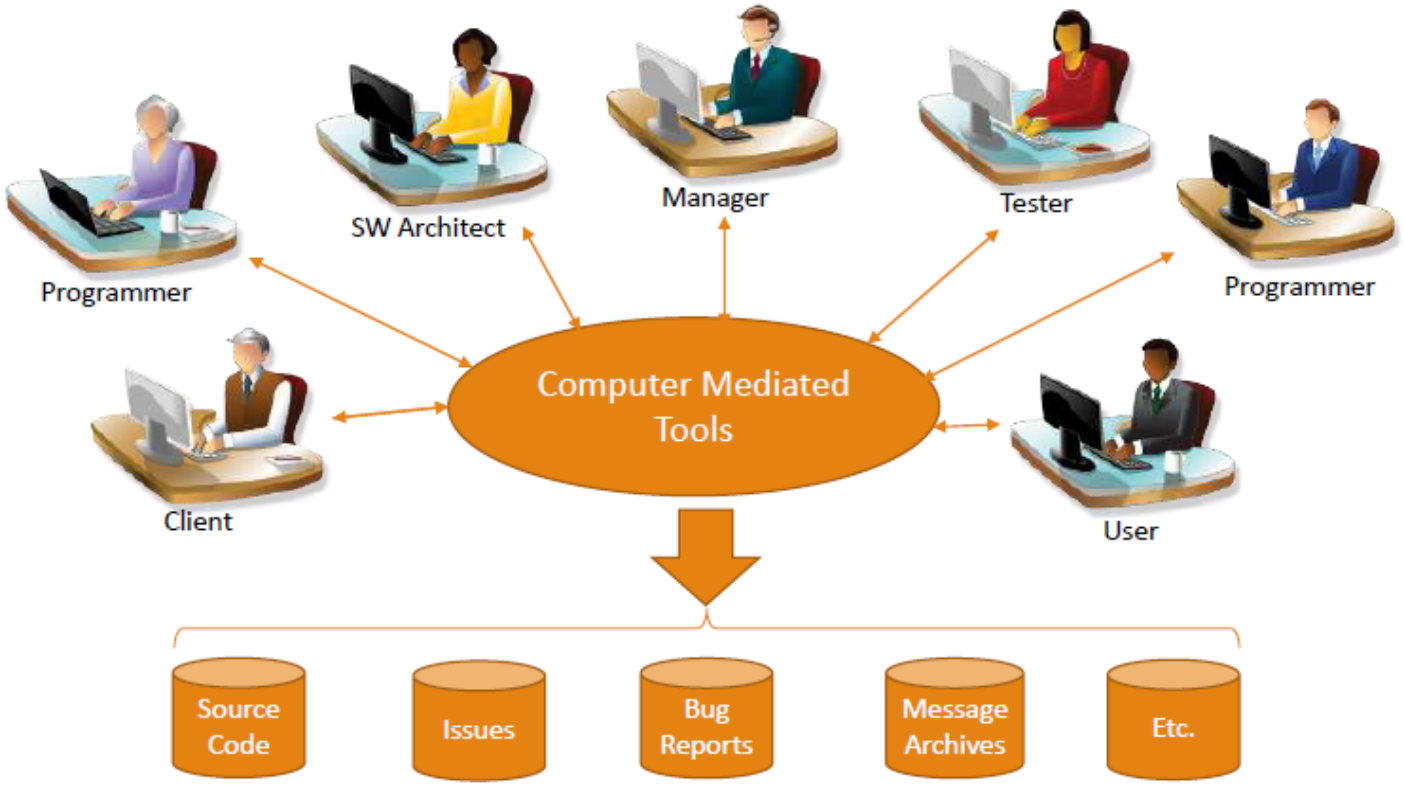
Lecture Goals

- **Learn about:**
  - Classic and notable research and researchers in mining 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
  - **Some of my work**
- **After the lecture, you should be able to:**
  - Retrieve SE data
  - Prepare SE data for mining
  - 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

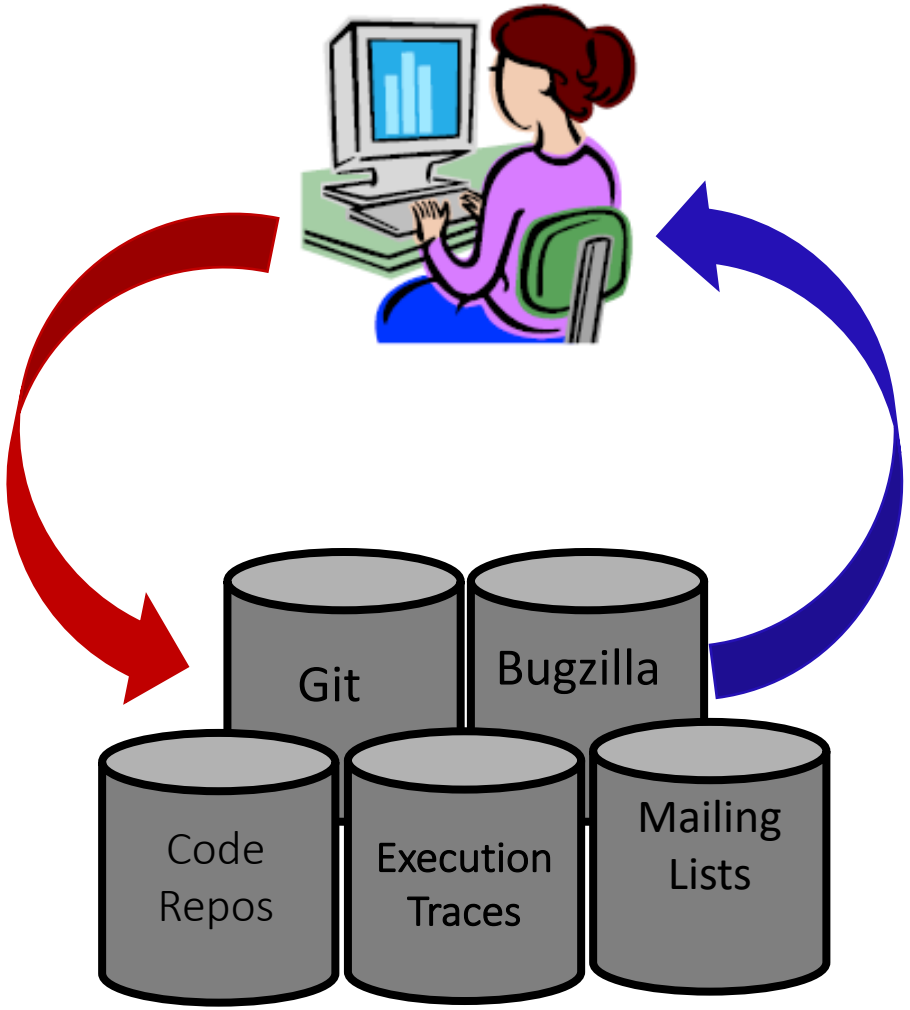
How is SE Data generated?



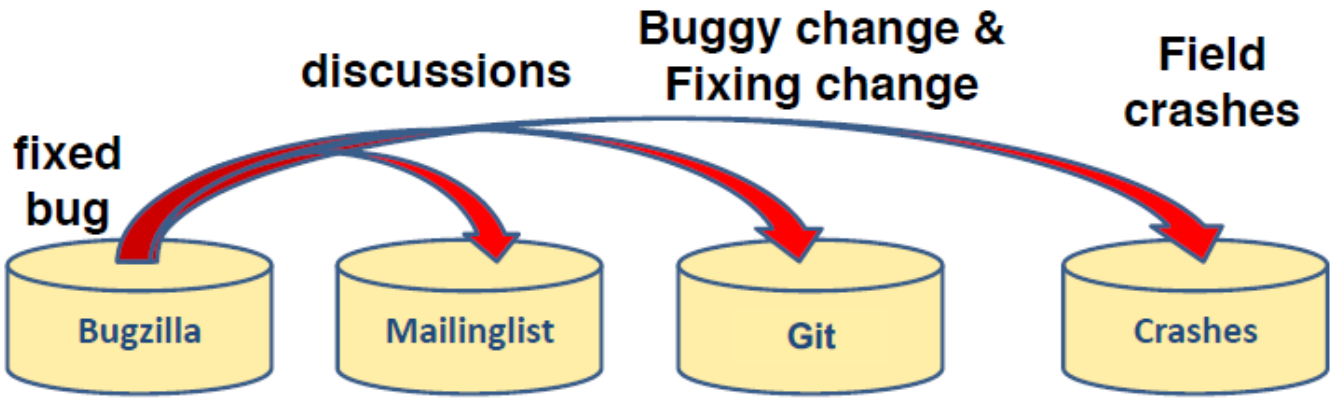
Current and historical artifacts and interactions are registered in software repositories

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



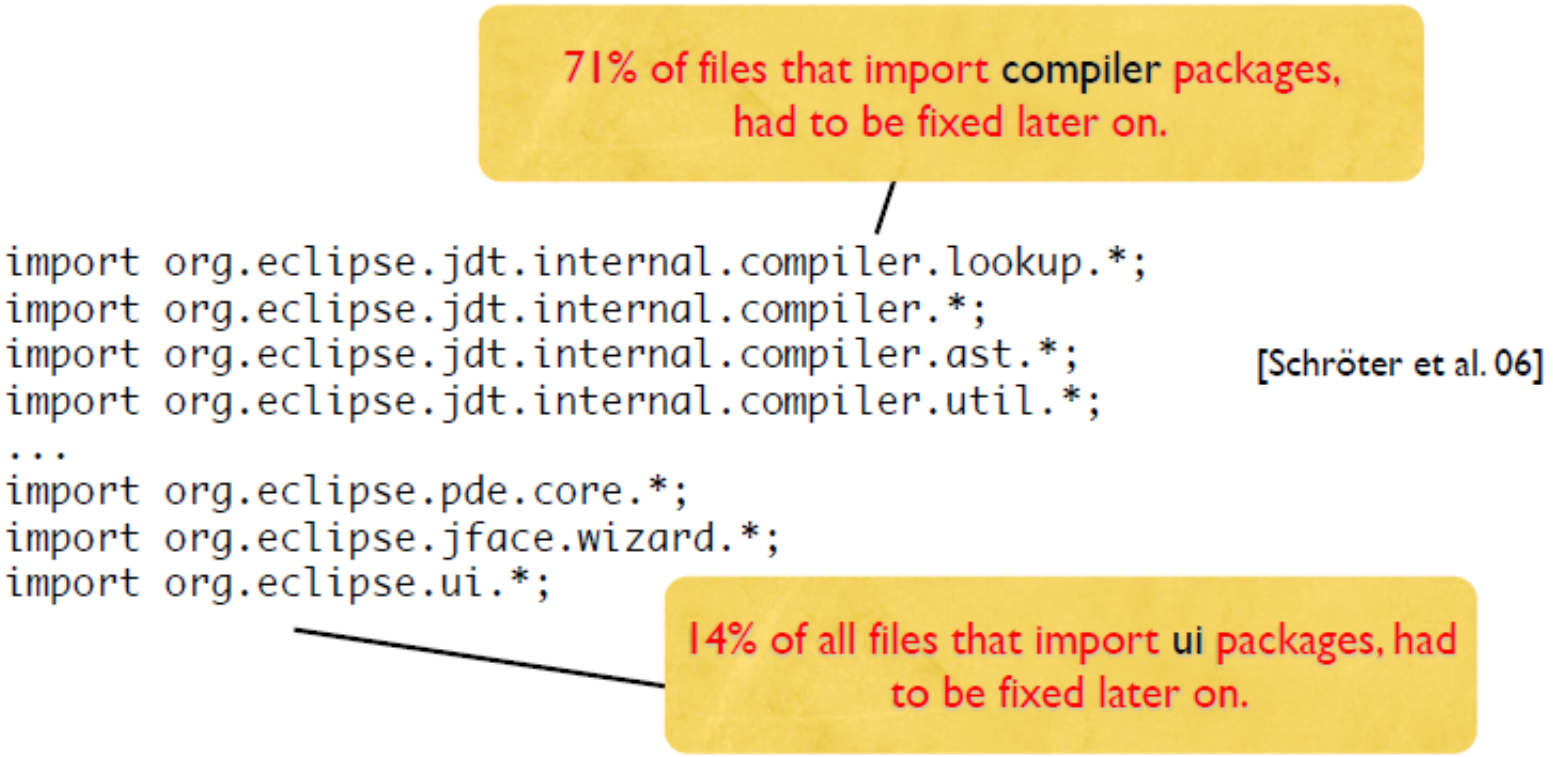
New bug report
Estimate fix effort
Suggest experts and fix!

Study Outline

- **Part I**: What can we learn from SE data?
  - A sample of notable findings for different SE data types
- **Part II**: How can we mine SE data?
  - Understand the structure of SE data

MSR studies – Bugs – Part I

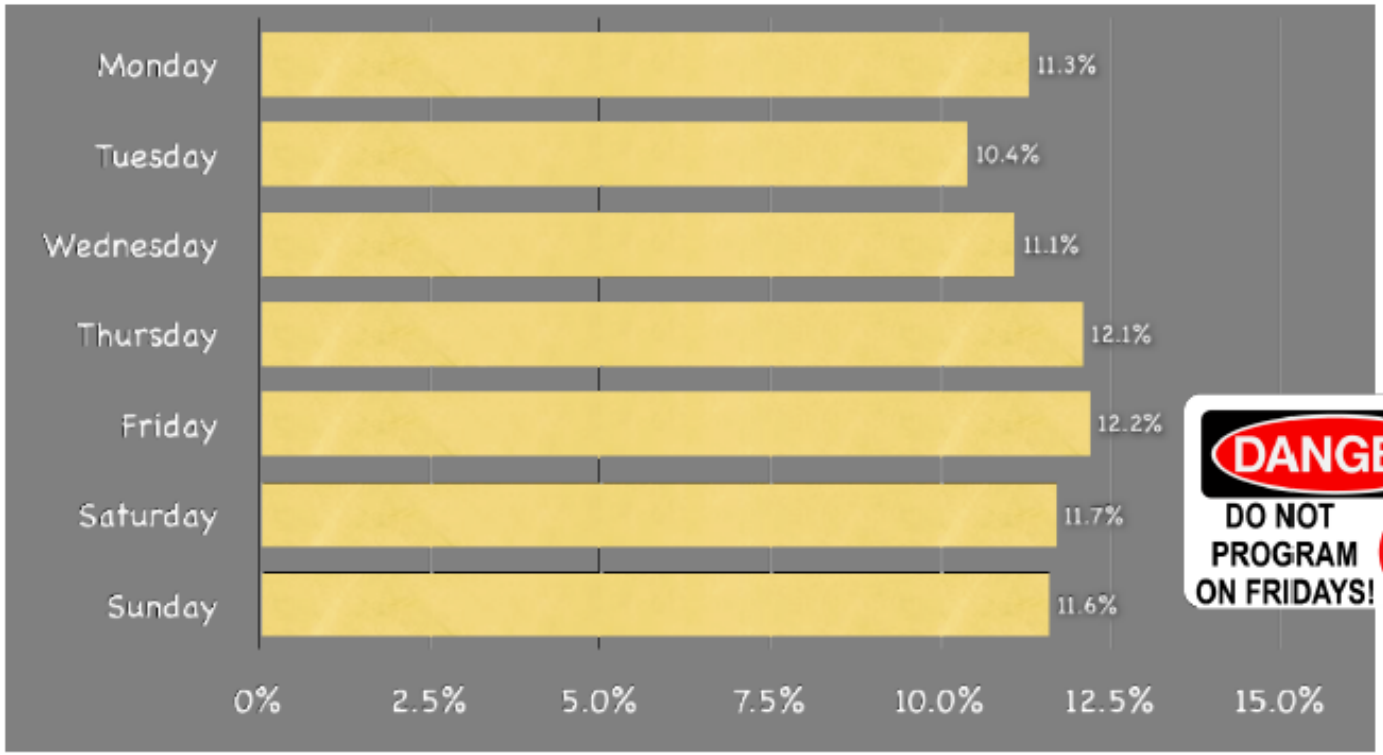
Using imports to predict Bugs



[Schröter et al. 06]

MSR studies - Bugs

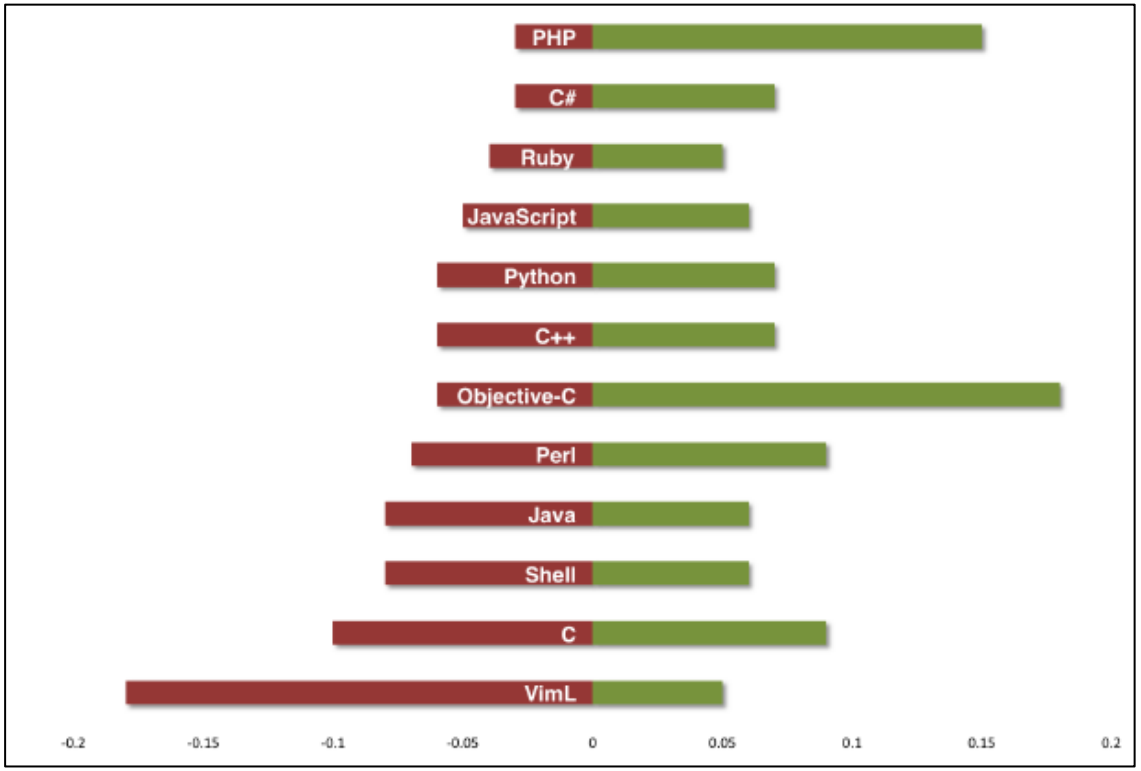
Do not program on Friday ;-)



Percentage of bug-introducing changes for eclipse [Zimmermann et al. 05]

MSR studies – Sentiment Analysis

Anger vs. Joy



How they stack up?

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

[Doll and Grigorik, 2012]

MSR studies – Sentiment Analysis

[redacted]

10/23/12 3:56 AM

Disable 'showmatch' option Matching parens are highlighted even without this option; what it does is jump the cursor to the matching paren which is [redacted] insane.

10/23/12 3:28 AM

styles everywhere, tutorial for first user login, fixing some css [redacted]

10/23/12 2:57 AM

[redacted] again

10/23/12 2:30 AM

more [redacted]

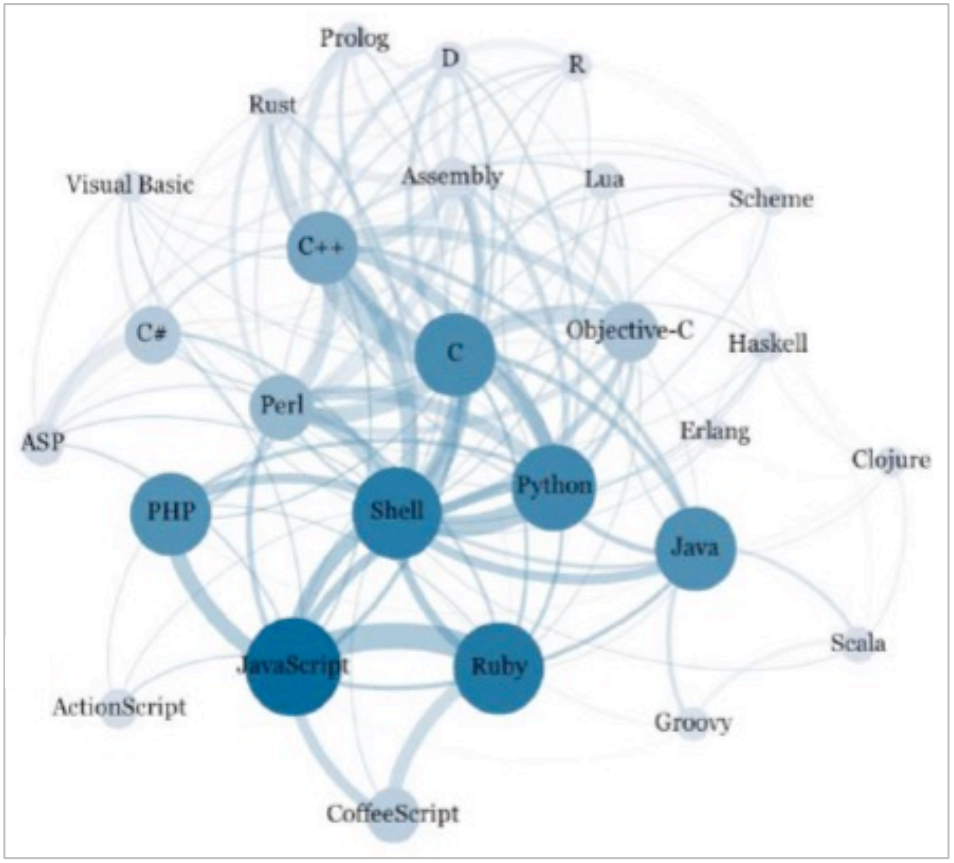
10/23/12 2:29 AM

Security [redacted] worked out

http://www.commitlogsfromlastnight.com/

MSR studies – Programming languages

Programming language relations



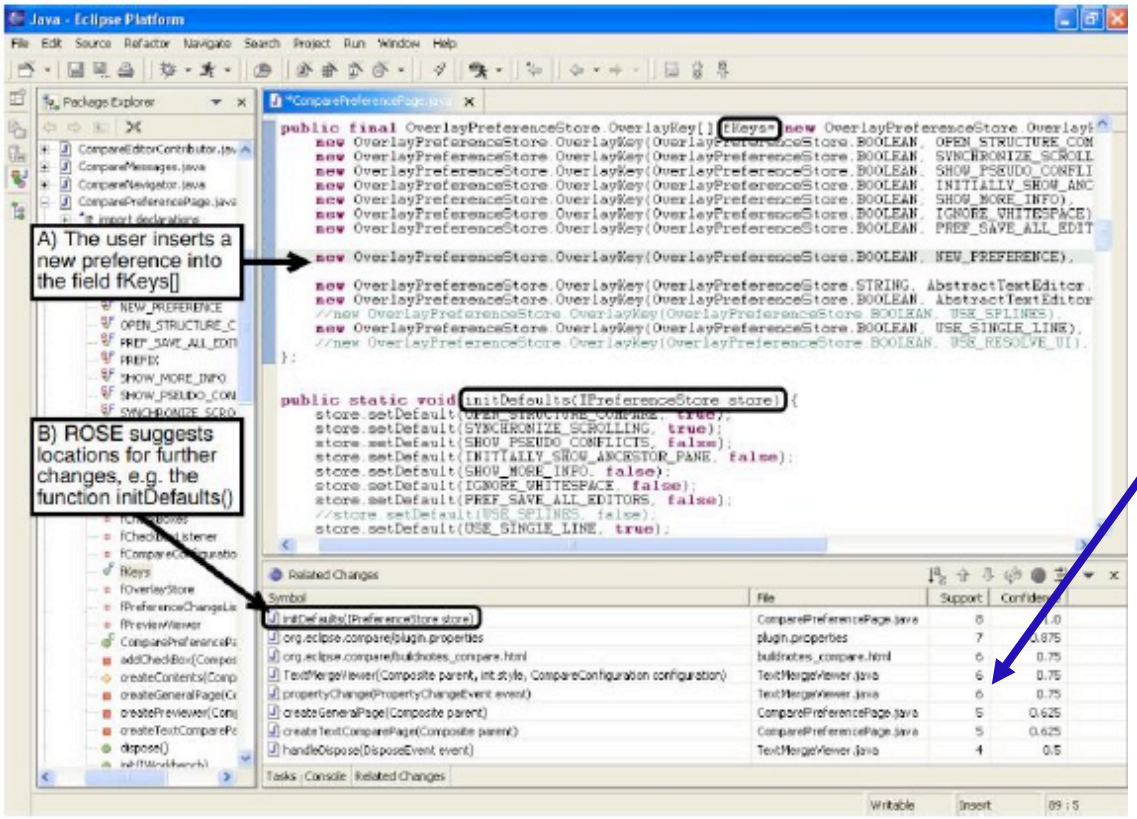
A **Ruby** programmers **is very likely to know Javascript**, while a **Perl** programmer is not

Java is a popular programming language but stands primarily alone

https://github.com/mjwillson/ProgLangVisualise

MSR studies – Changes by programmers

Programming language relations



After the programmer has made some changes to the source (above), **ROSE** suggests locations (below) where, in similar transactions in the past, further changes were made

[Zimmermann, 2005]
Mining Version Histories to Guide Software Changes

How can we mine SE Data – Part II

Repositories of Repositories

January 2020:
100 Millinon repositories
40 Million Users

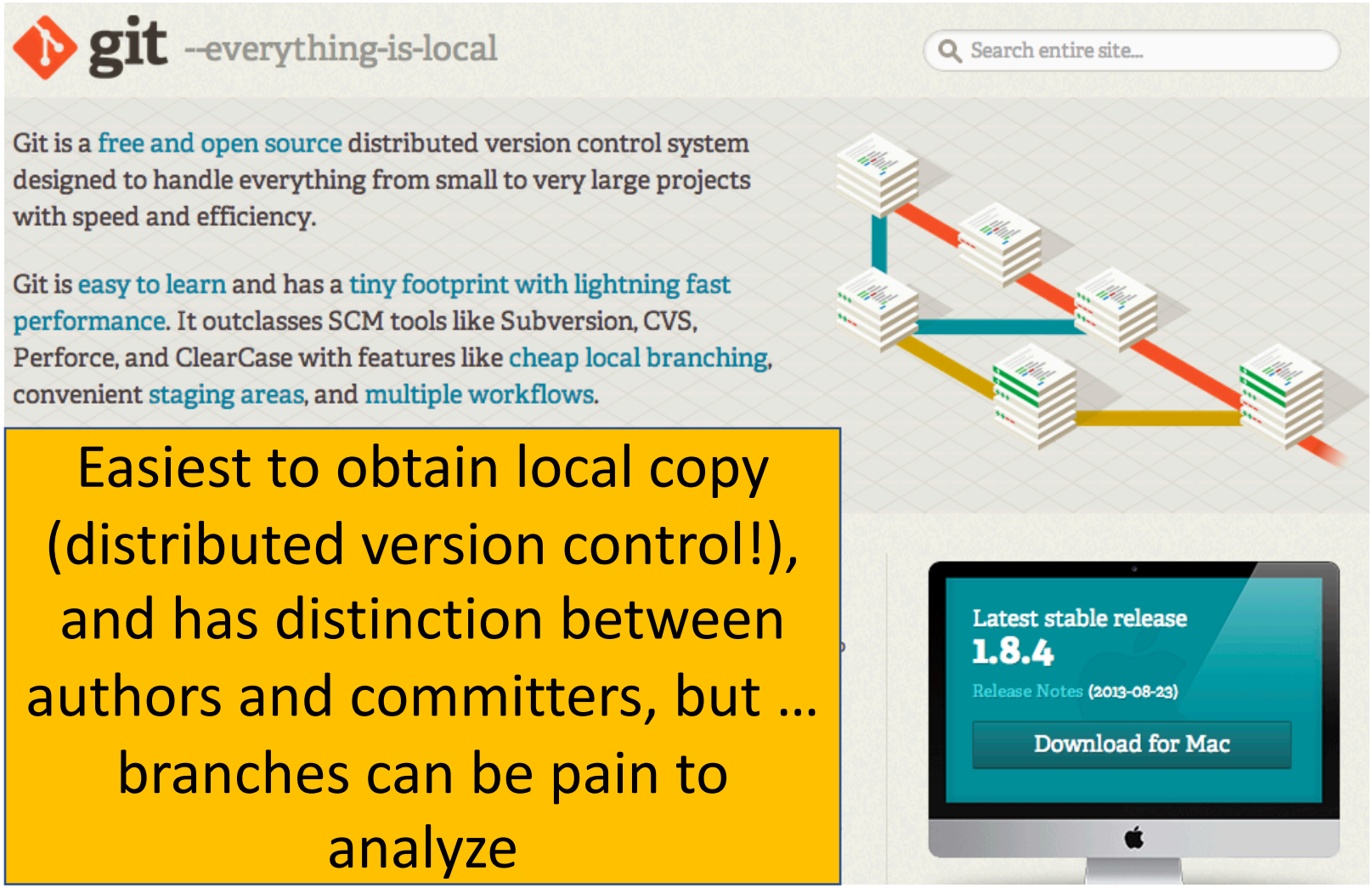
April 2019
28 Millinon repositories
10 Million Users

April 2019
28K projects

January 2020:
430K repositories
3.7 Million Users

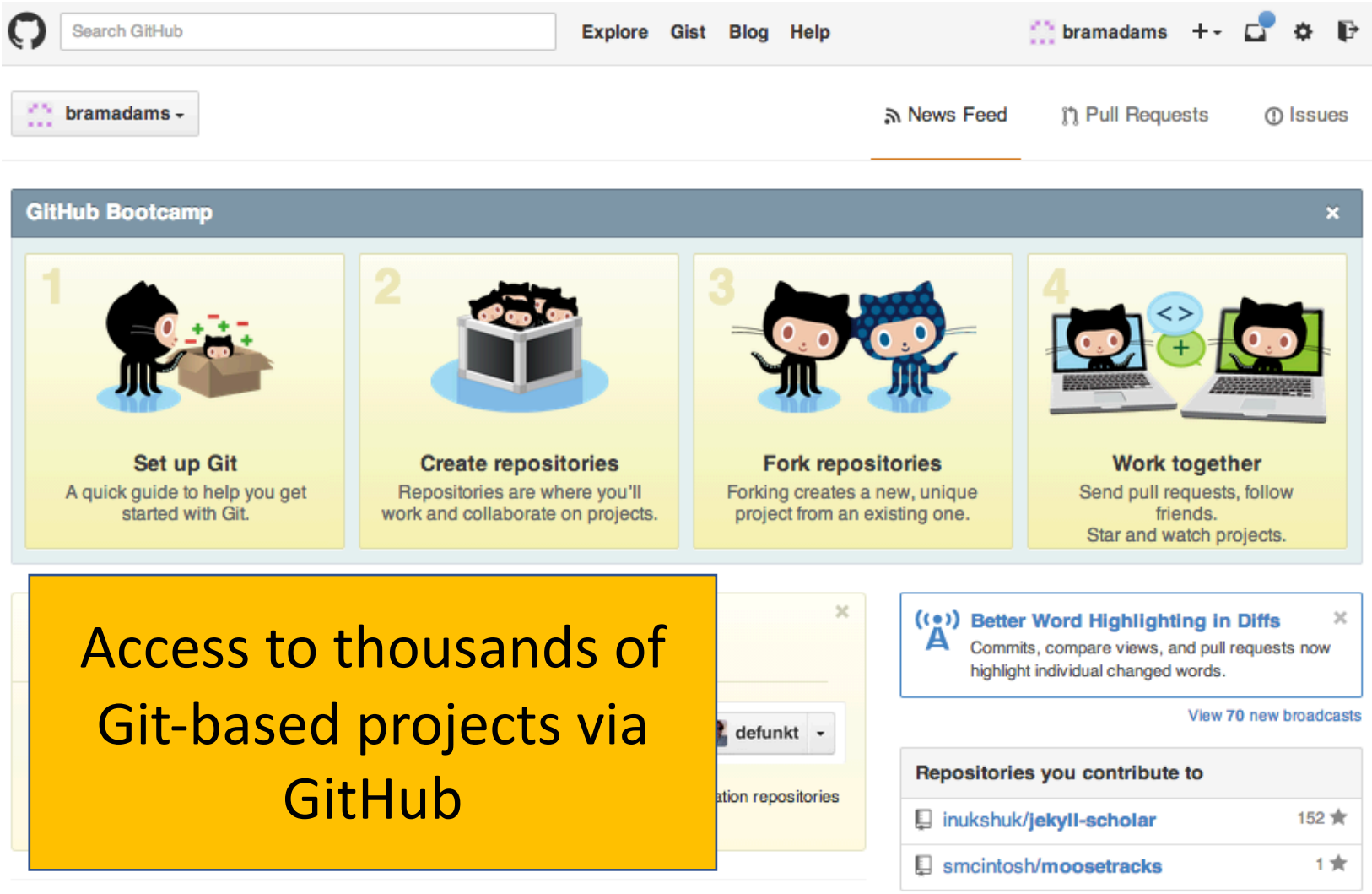
How can we mine SE Data

How can we mine SE Data

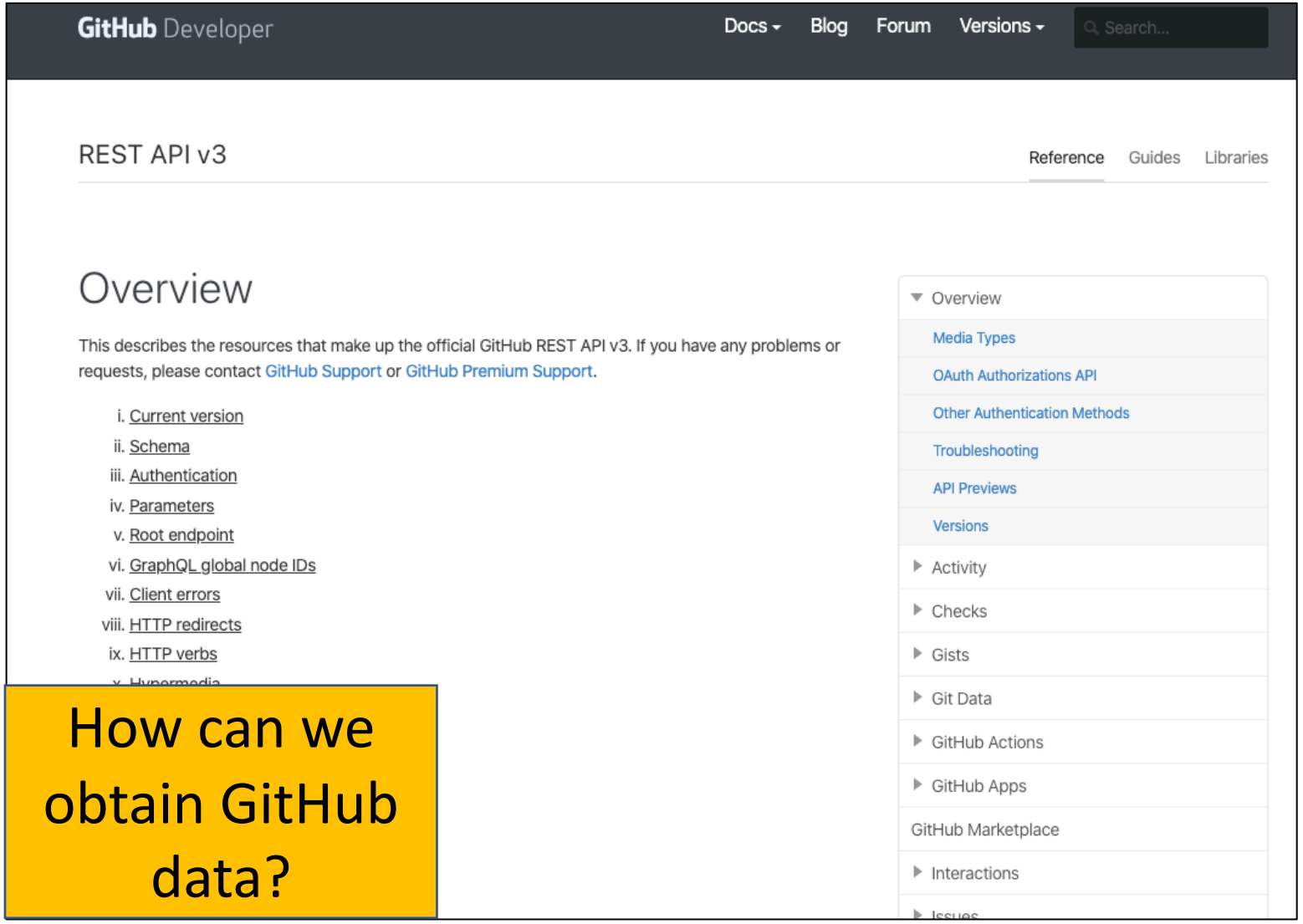


Easiest to obtain local copy (distributed version control!), and has distinction between authors and committers, but … branches can be pain to analyze

How can we mine SE Data

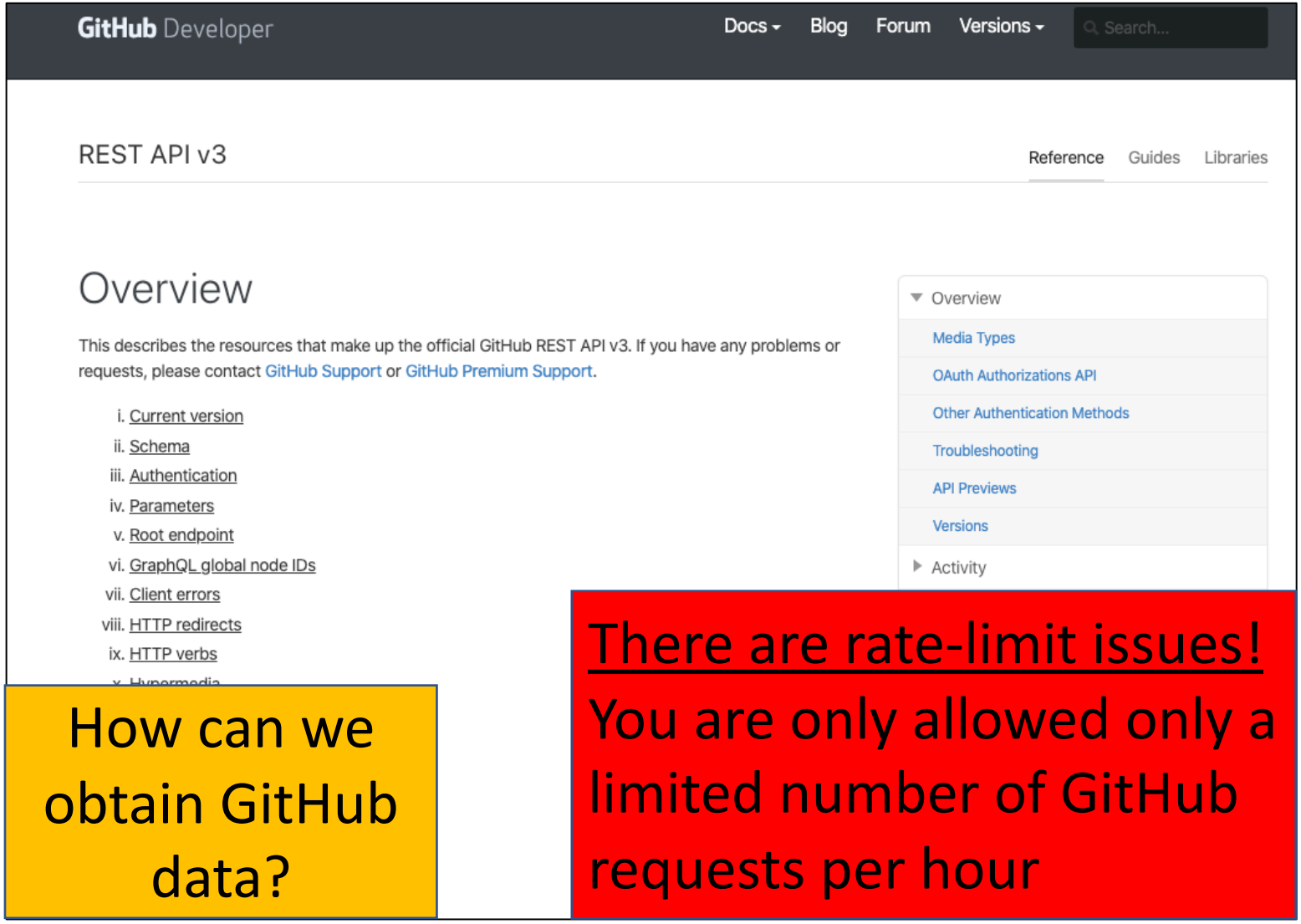


How can we mine GitHub Data



How can we obtain GitHub data?

How can we mine GitHub Data

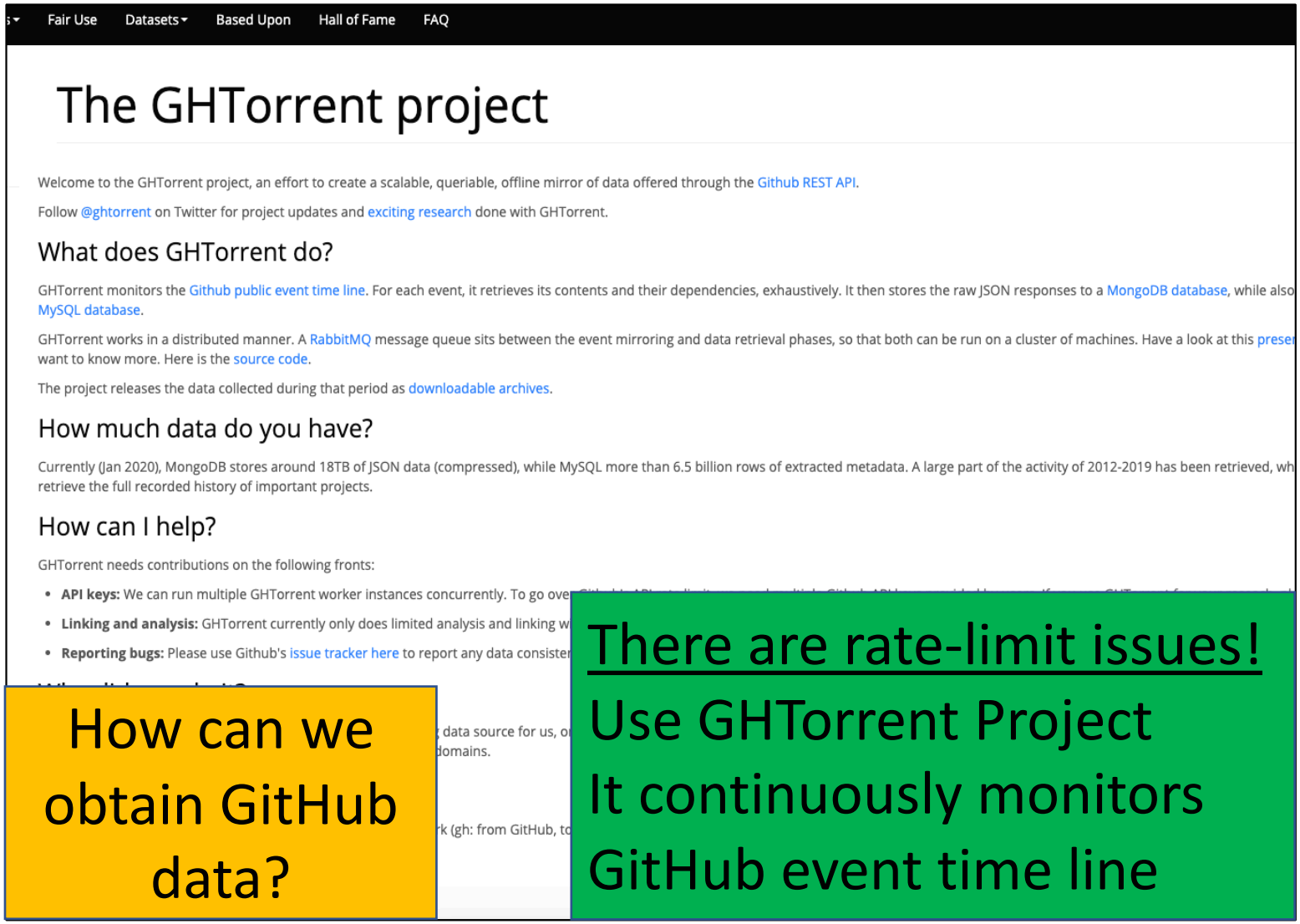


How can we obtain GitHub data?

There are rate-limit issues!

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

How can we mine GitHub Data



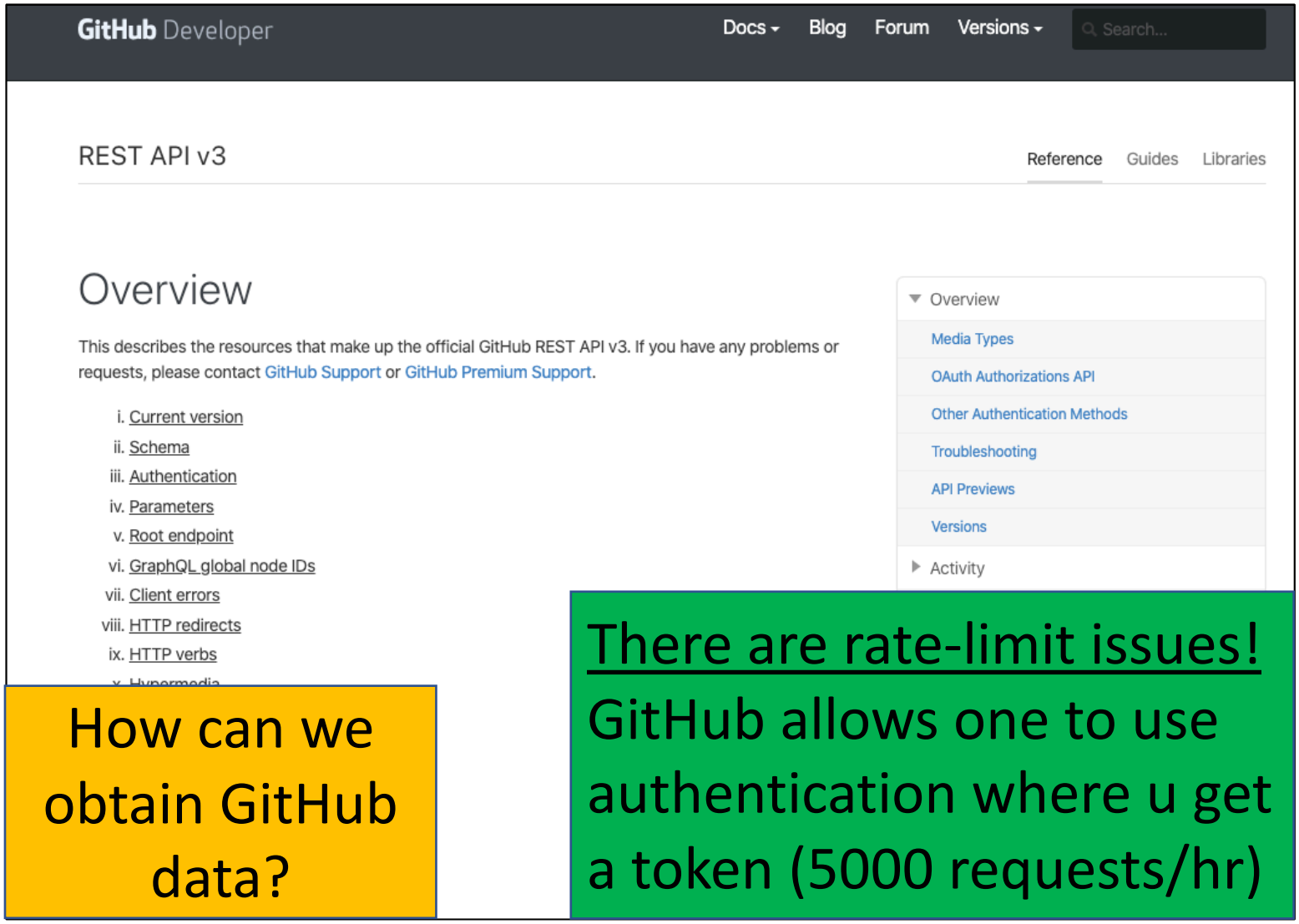
How can we obtain GitHub data?

There are rate-limit issues!

Use GHTorrent Project

It continuously monitors GitHub event time line

How can we mine GitHub Data



How can we obtain GitHub data?

There are rate-limit issues!

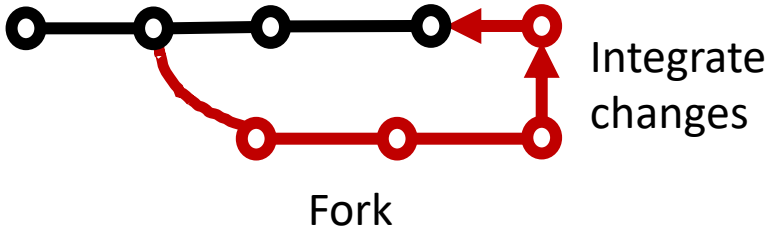
GitHub allows one to use authentication where u get a token (5000 requests/hr)

Some of my work

Patch Propagation in Related Applications

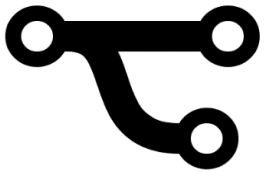
John Businge
John.businge@uantwerpen.be

Introduction



Reuse in Android ecosystem

Nov - 2019
>940K repositories,
48K users

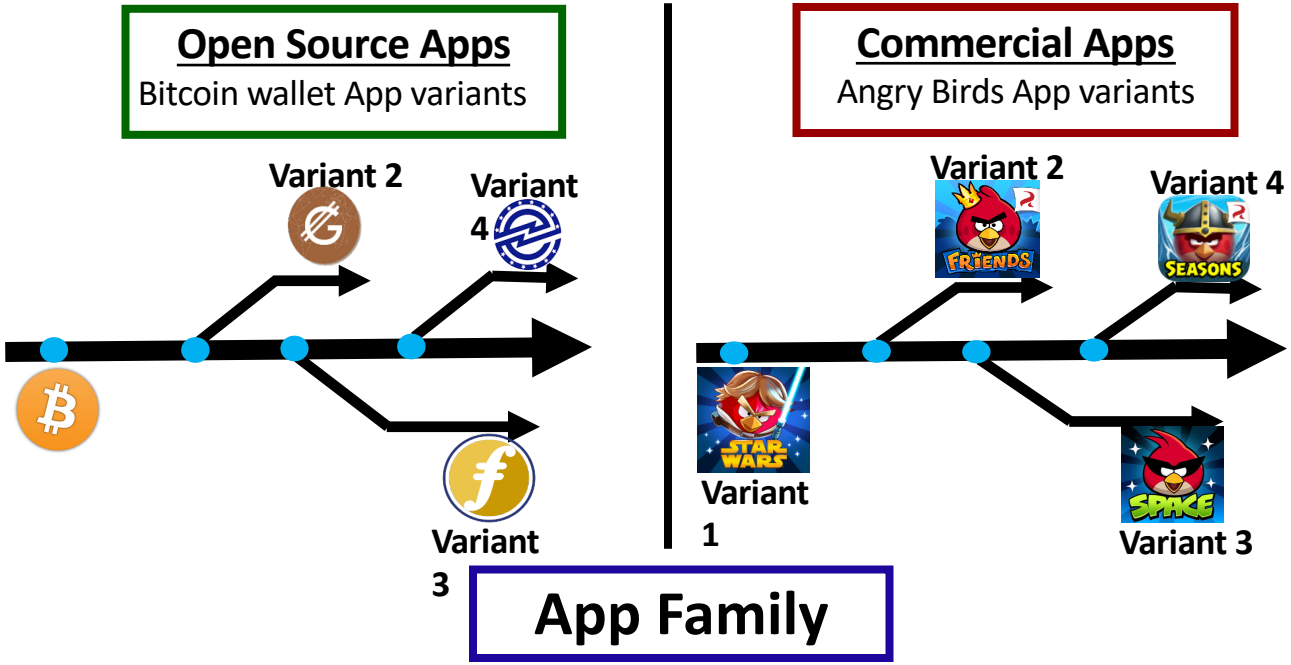


>3K Android repos
>100 forks

Software variants

J. Businge, et al., "Clone based variability management in the android ecosystem", *ICSME*, 2018.

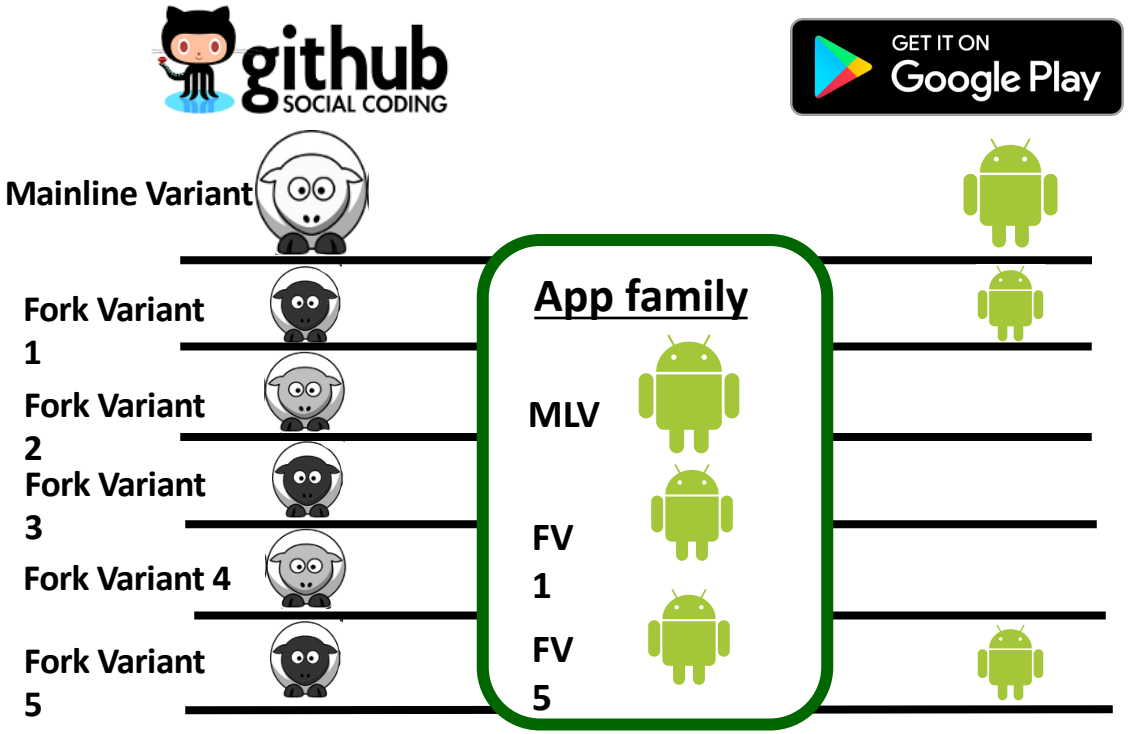
Android Mobile-app variants



Reasons for creating App Variants:

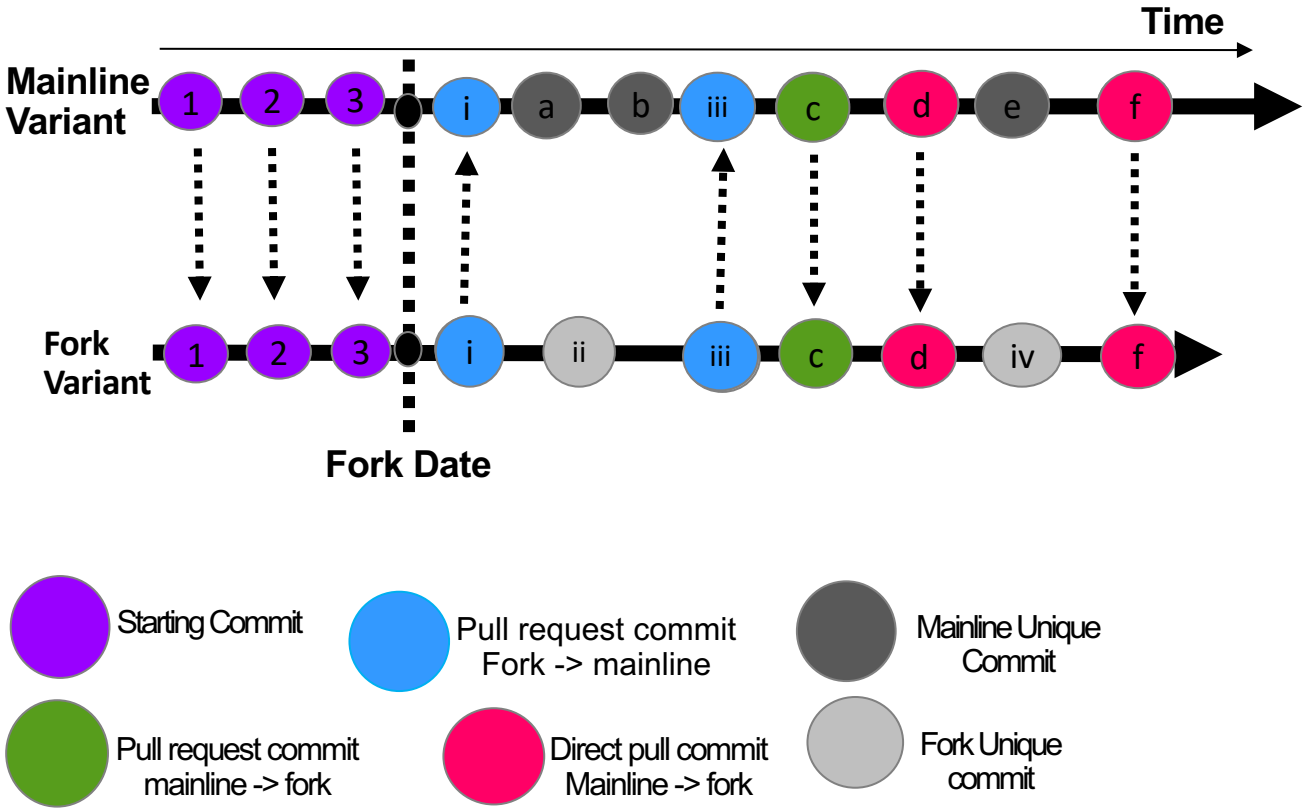
- Different user needs
- Different markets
- Different payment modes

App Family Identification

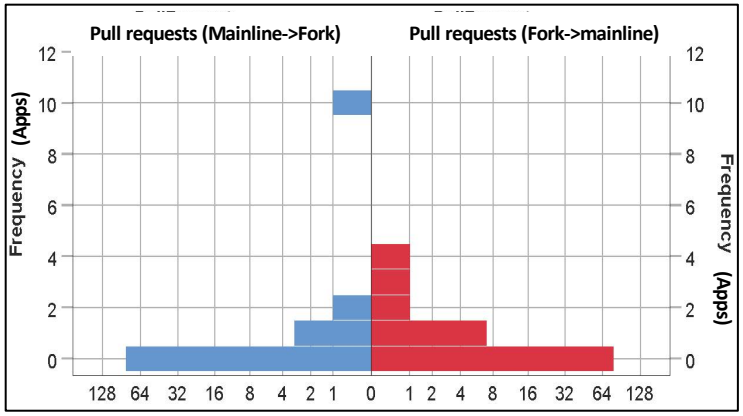


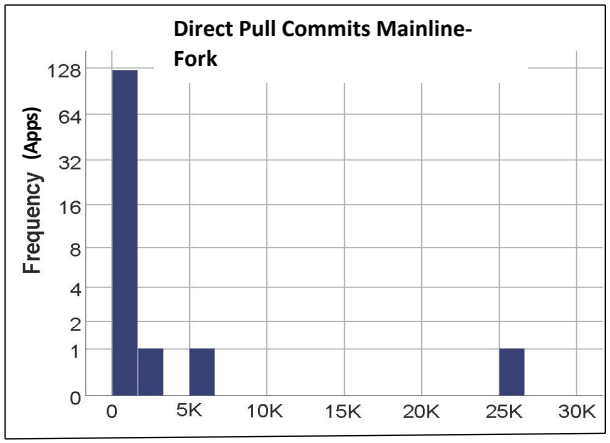
J. Businge, et al., "Clone based variability management in the android ecosystem", *ICSME*, 2018.

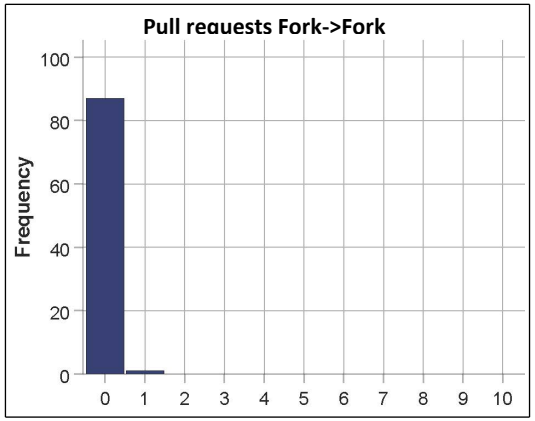
Code propagation in the Android Family



Code Propagation Results – ICSME 2018



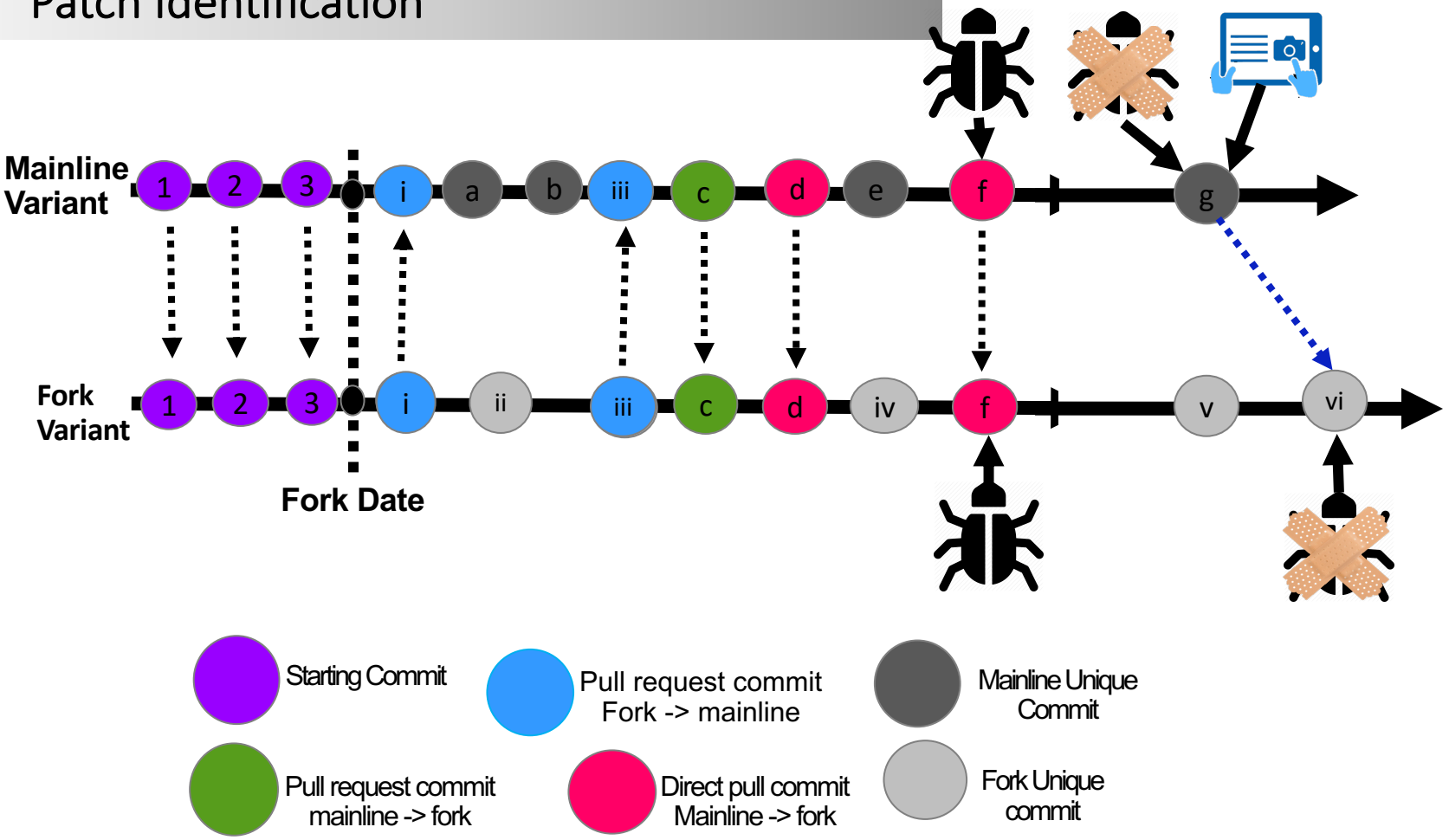




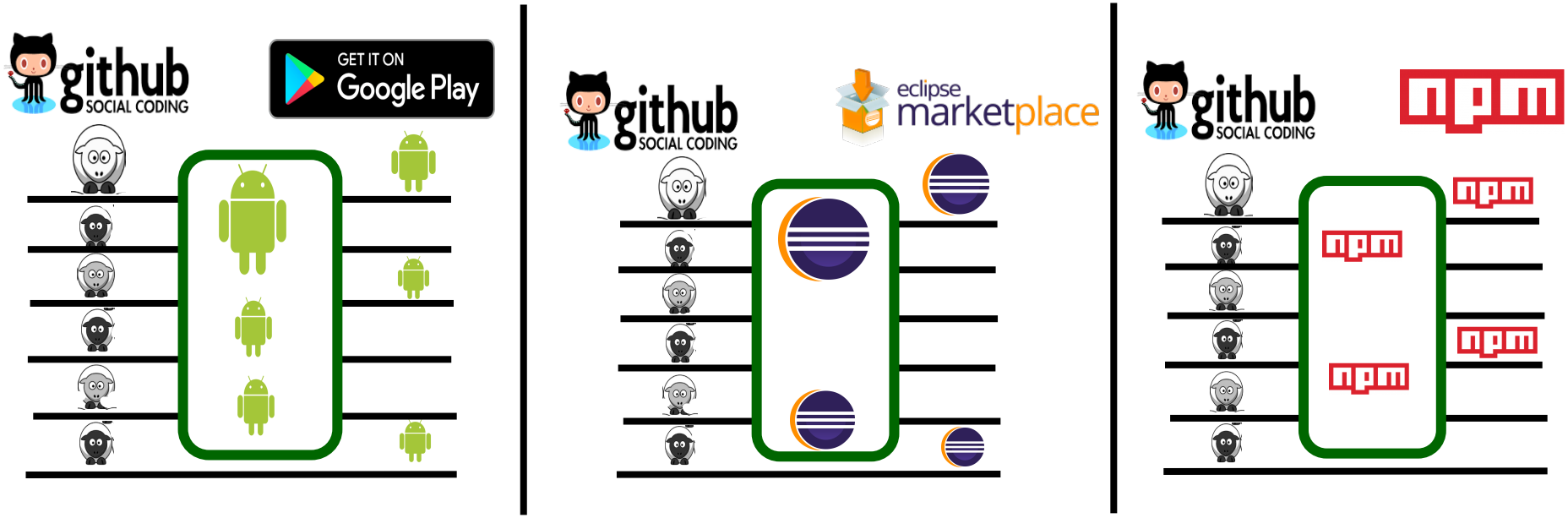
- Limited code propagation between variants
- Developer's difficulty of finding patches in shared artifacts
- GitHub only provides how many commits a variant is ahead/behind w.r.t another

J. Businge, et al., "Clone based variability management in the android ecosystem", *ICSME*, 2018.

Patch Identification

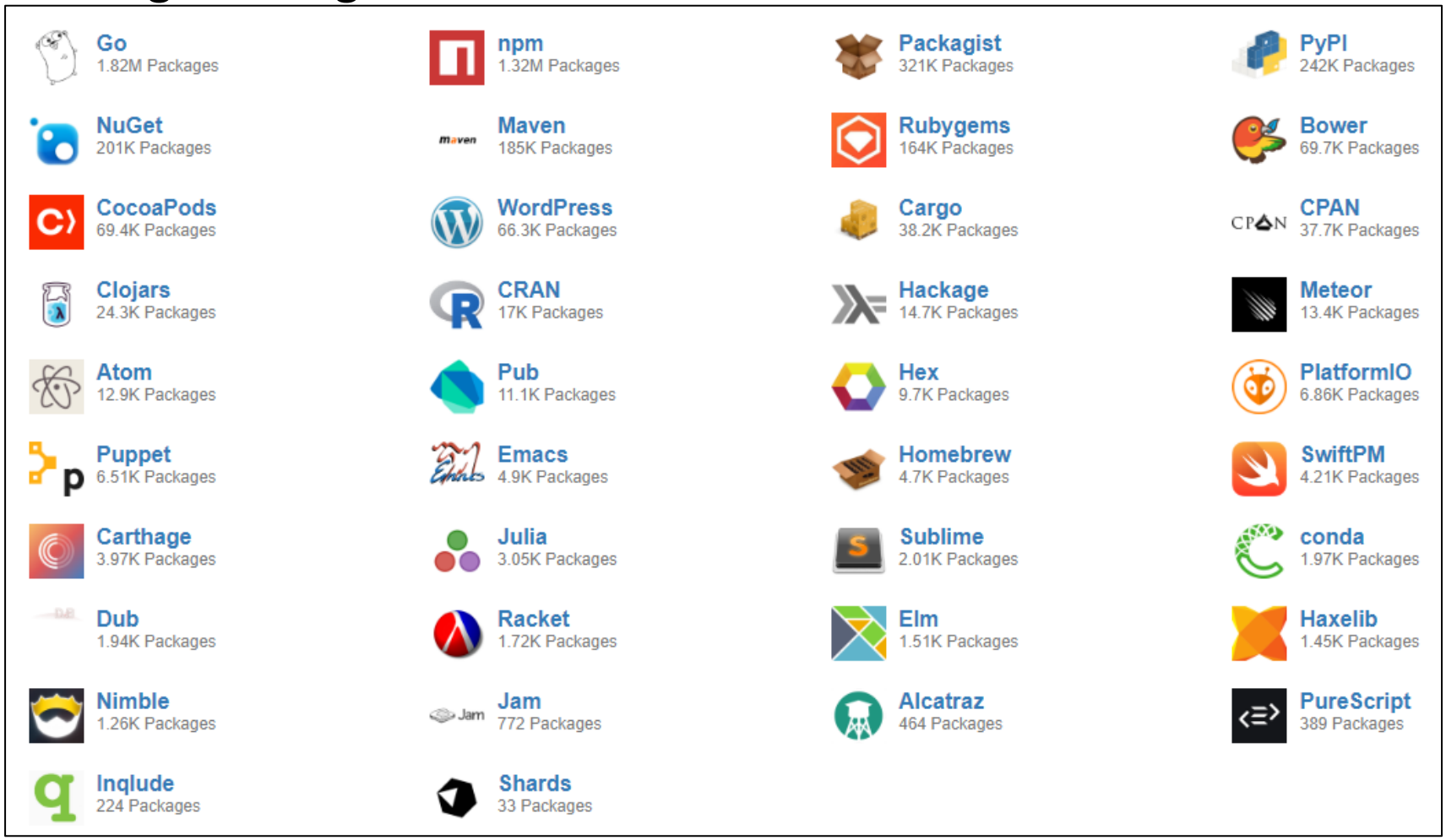


Method 1 - Collect Study Subjects

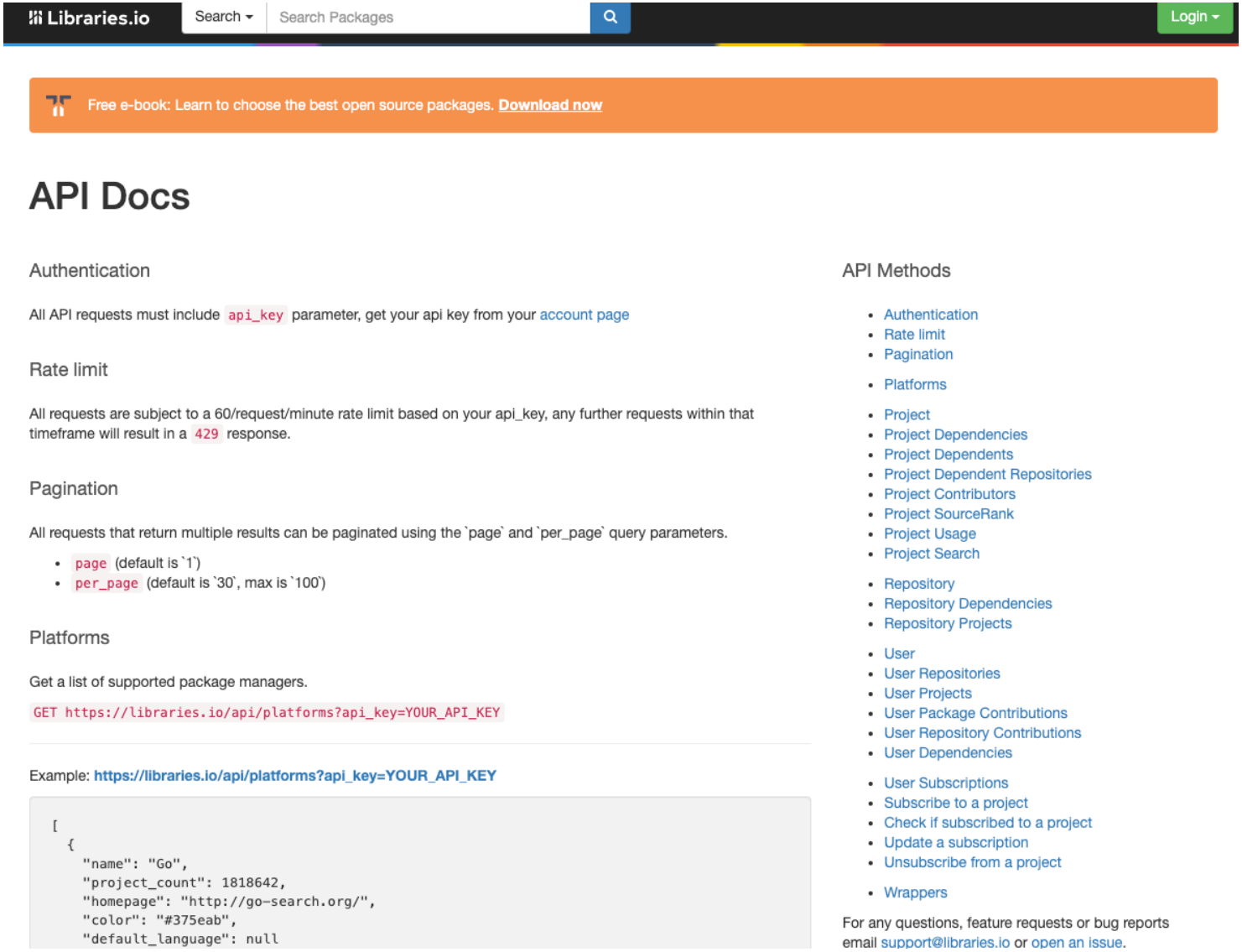


How can we mine SE Data – Part II

Package managers - Libraries.io



How can we mine SE Data from package managers?



Libraries.io Search Search Packages Login

Free e-book: Learn to choose the best open source packages. Download now

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 or open an issue.

Let’s get our hands dirty