

Mining Software Repositories

John Businge
Henrique Rocha



References

The Road Ahead for Mining Software Repositories

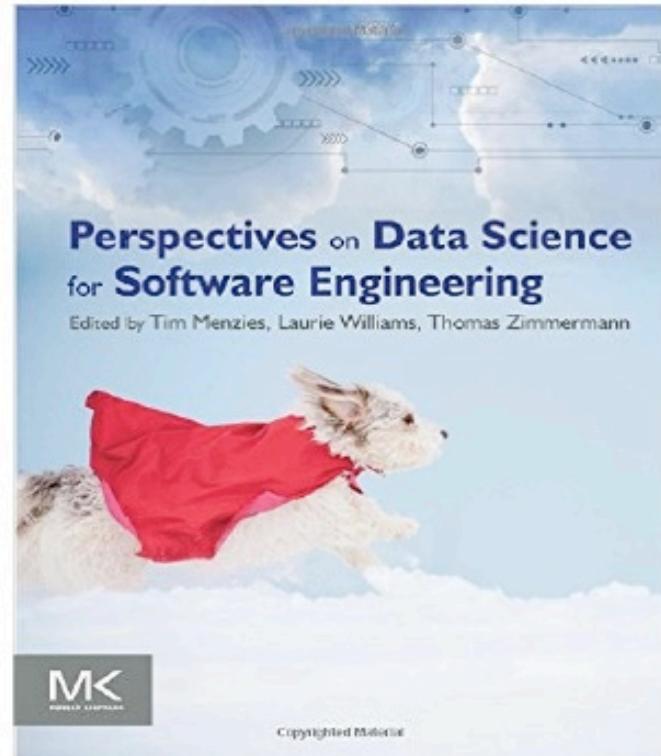
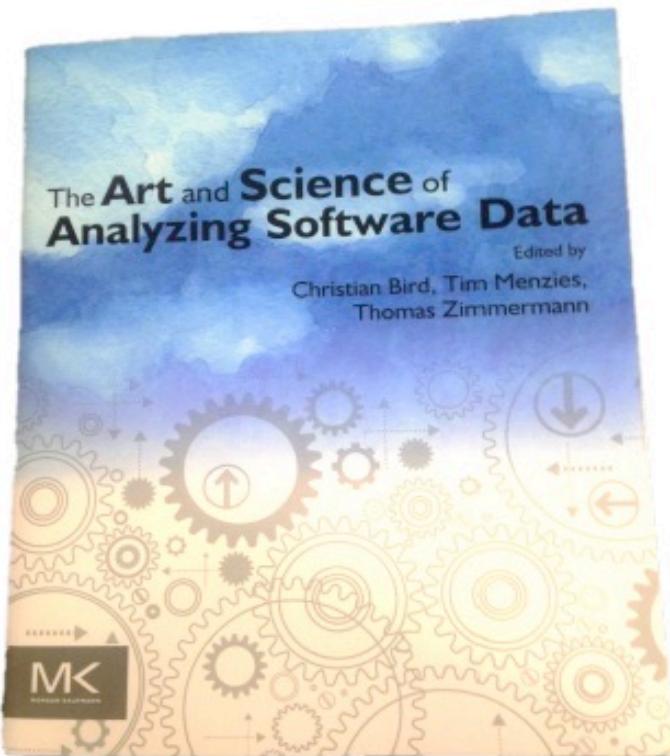
Ahmed E. Hassan
Software Analysis and Intelligence Lab (SAIL)
School of Computing, Queen's University, Canada
ahmed@cs.queensu.ca

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

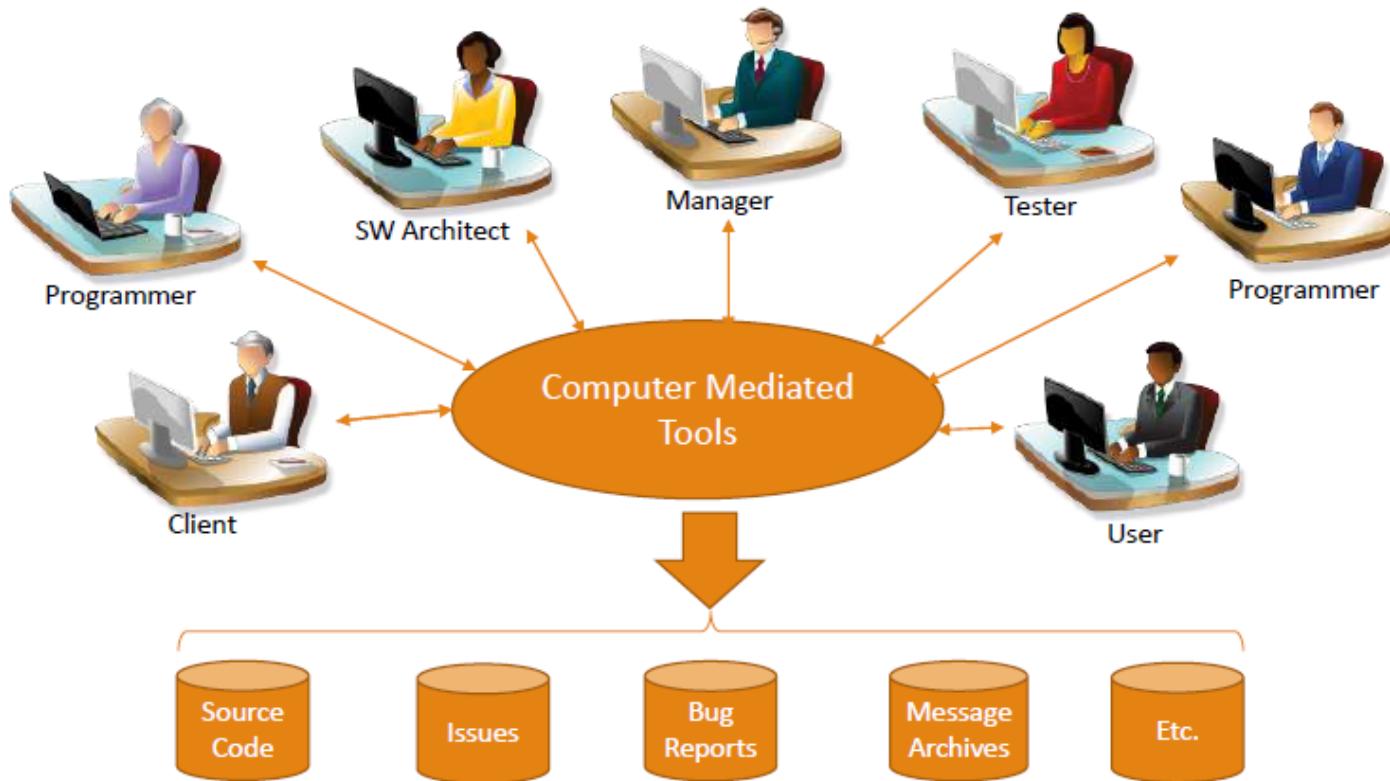
Why mine SE data?

- **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



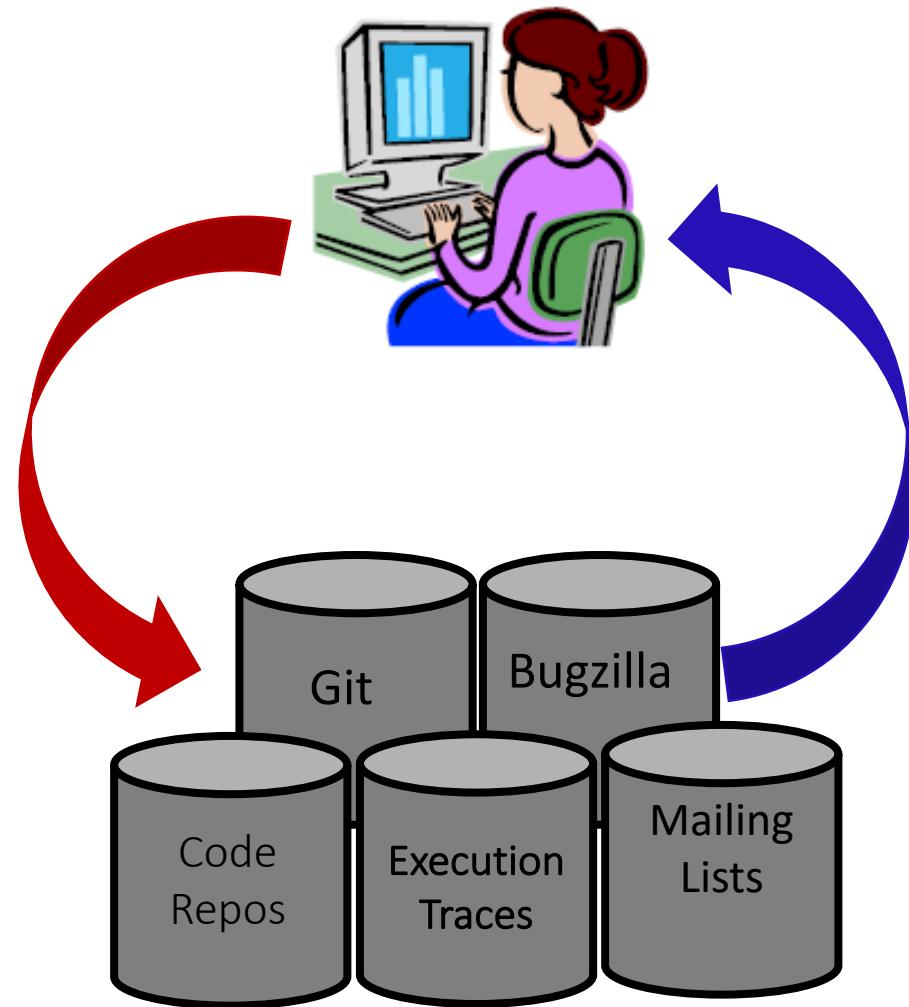
How is SE Data generated?



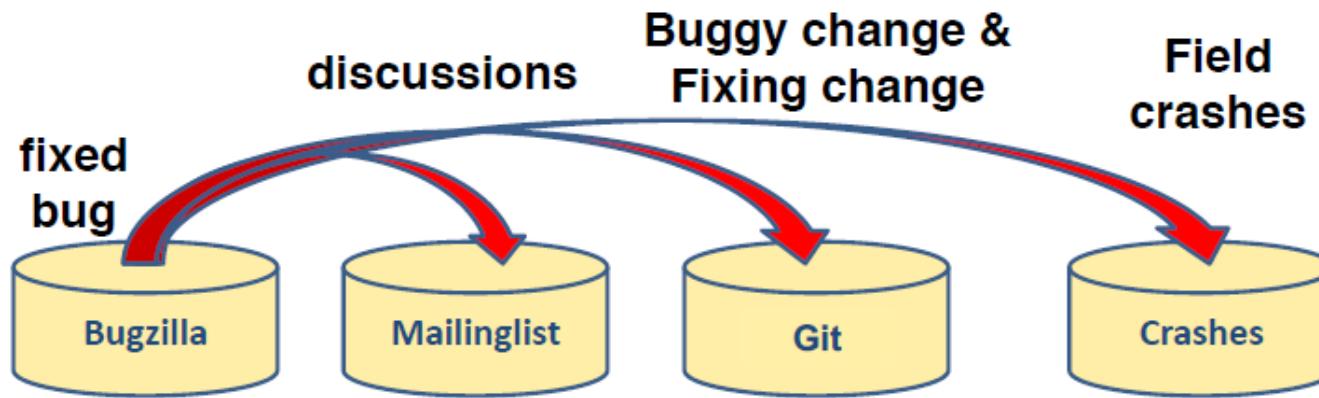
Current and historical artifacts and interactions are registered in software repositories

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

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

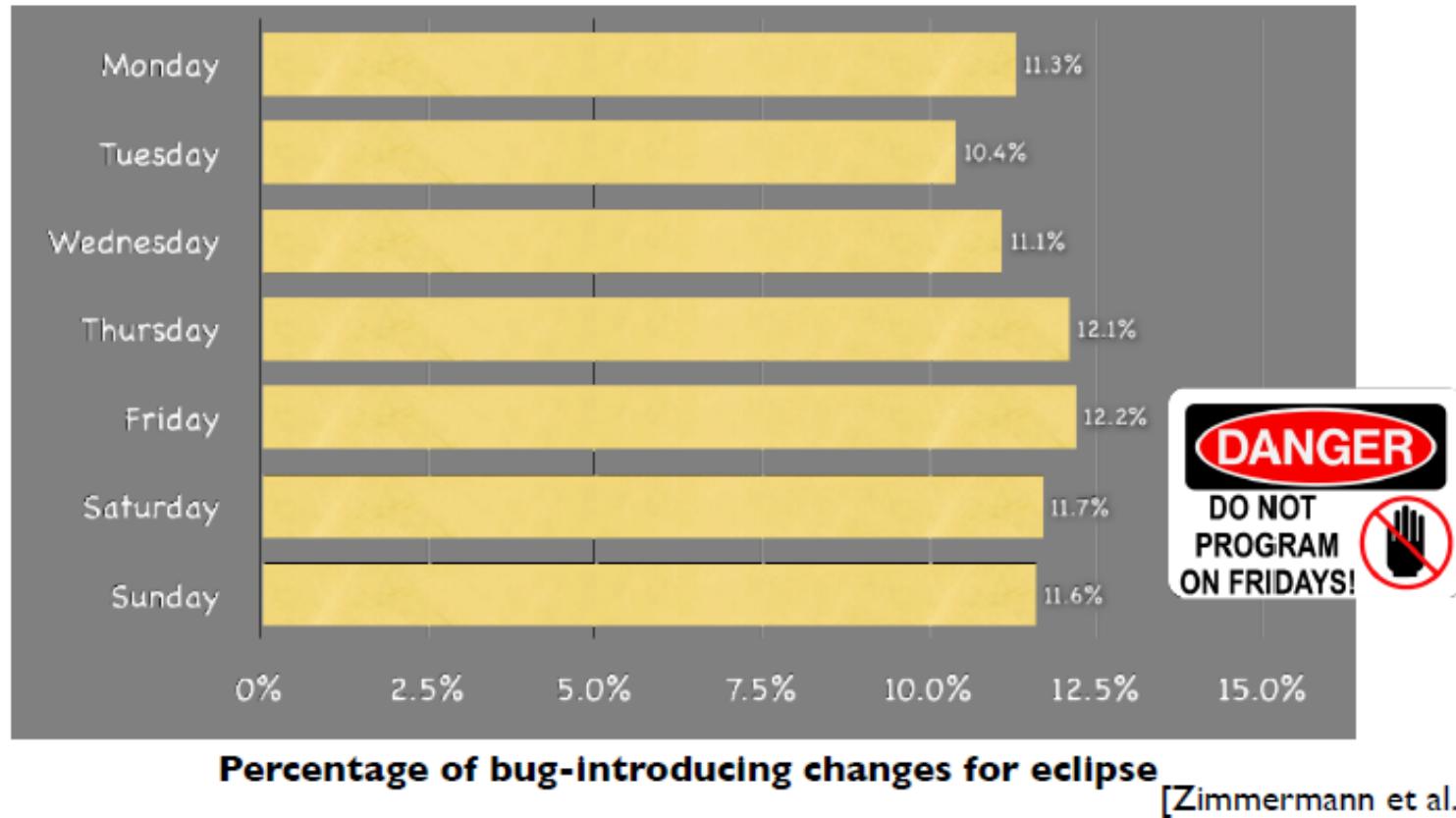
```
import org.eclipse.jdt.internal.compiler.lookup.*;  
import org.eclipse.jdt.internal.compiler.*;  
import org.eclipse.jdt.internal.compiler.ast.*;  
import org.eclipse.jdt.internal.compiler.util.*;  
...  
import org.eclipse.pde.core.*;  
import org.eclipse.jface.wizard.*;  
import org.eclipse.ui.*;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

[Schröter et al. 06]

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

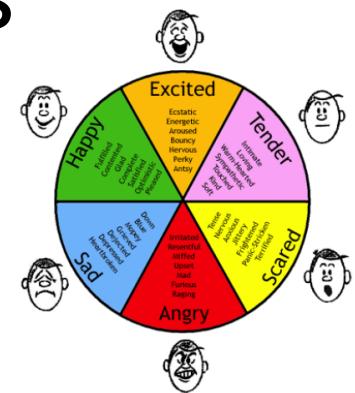
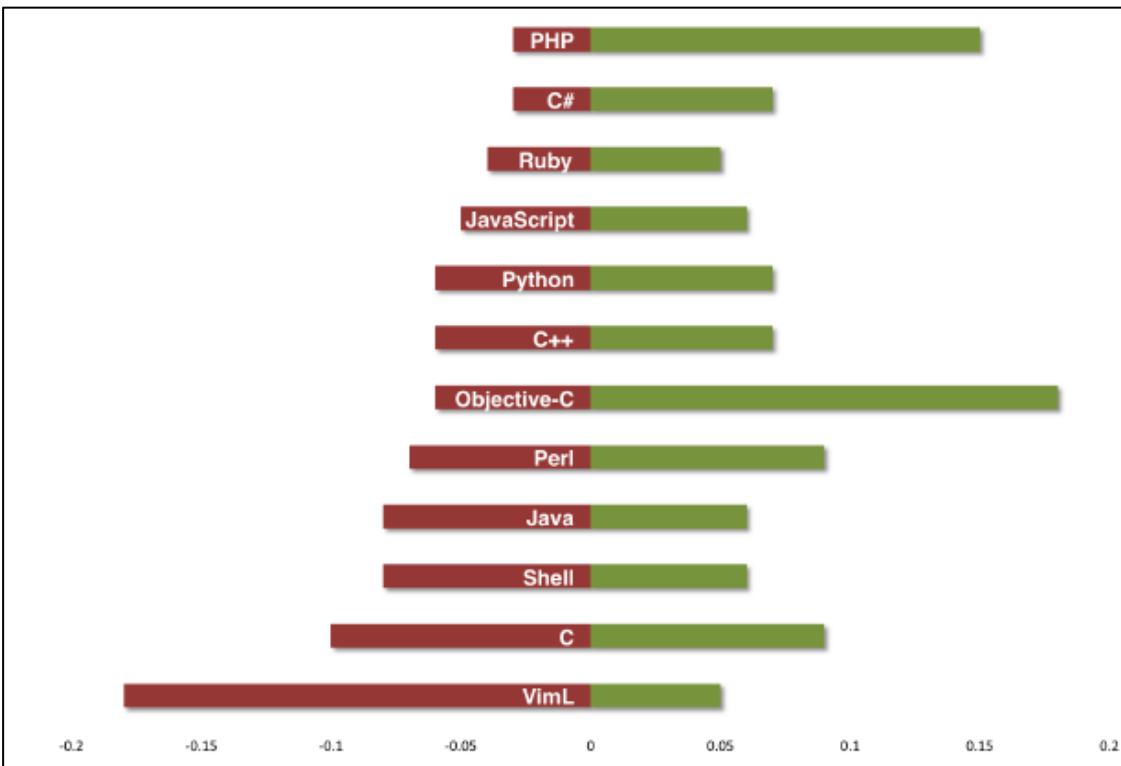
MSR studies - Bugs

Do not program on Friday ;-)



MSR studies – Sentiment Analysis

Anger vs. Joy

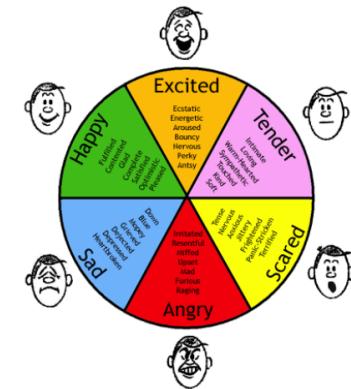


How they stack up?

- **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



-  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

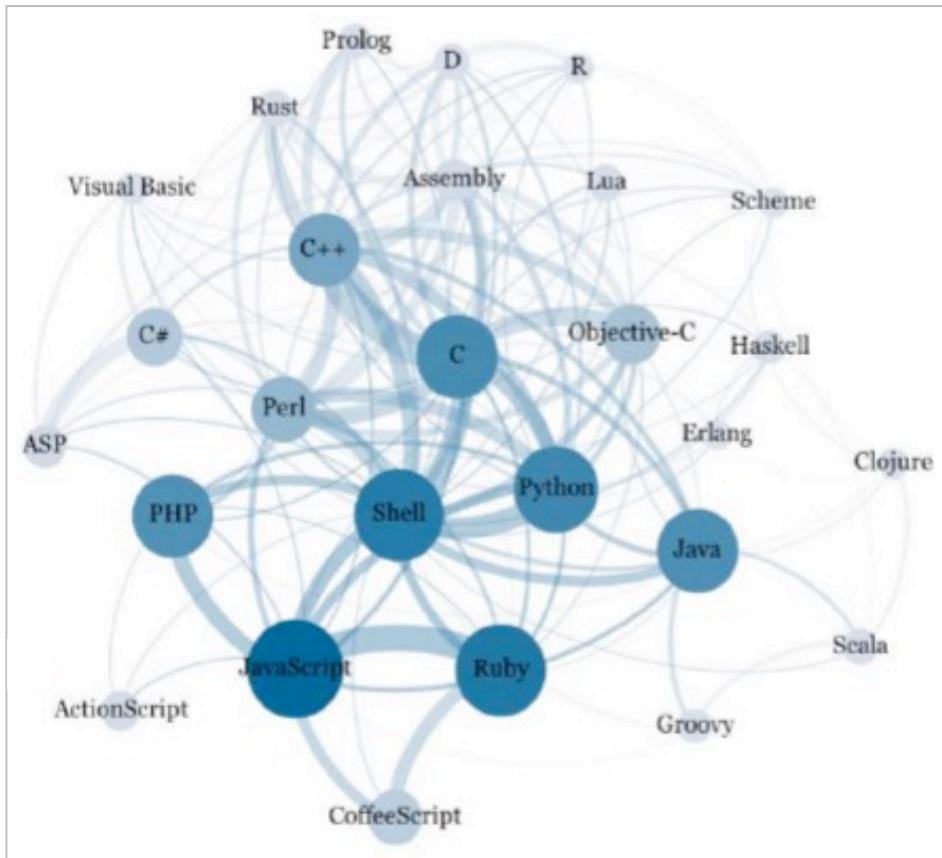
 10/23/12 2:30 AM more [REDACTED]

 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

The screenshot shows the Eclipse IDE interface. On the left, the Package Explorer shows several Java files under the 'CompareEditorContributor' package. In the center, a code editor displays Java code related to OverlayPreferenceStore. Two annotations are present:

- A) The user inserts a new preference into the field fKeys[]**: An annotation with an arrow pointing to the line of code where a new key is being added to the fKeys array.
- B) ROSE suggests locations for further changes, e.g. the function initDefaults()**: An annotation with an arrow pointing to the declaration of the `initDefaults` function.

On the right, a 'Related Changes' view is open, showing a list of changes across various files, each with a confidence score. A blue arrow points from the 'ROSE suggests' annotation towards this view.

Symbol	File	Support	Confidence
initDefaults(IPreferenceStore store)	ComparePreferencePage.java	6	0.0
	plugin.properties	7	0.875
	buildnotes_compare.html	6	0.75
	TextMergeViewer.java	6	0.75
	TextMergeViewer.java	6	0.75
	ComparePreferencePage.java	5	0.625
	ComparePreferencePage.java	5	0.625
	TextMergeViewer.java	4	0.5

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 Million repositories
40 Million Users



January 2020:
430K repositories
3.7 Million Users



April 2019
28 Million repositories
10 Million Users



April 2019
28K projects



How can we mine SE Data



How can we mine SE 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 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

The screenshot shows the GitHub interface with a red bar at the top. Below it, the user 'bramadams' is logged in. The main content area displays the 'GitHub Bootcamp' tutorial, which consists of 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. To the right of the tutorial, there's a yellow box containing the text: 'Access to thousands of Git-based projects via GitHub'. Further down, there's a 'Better Word Highlighting In Diffs' feature box and a 'Repositories you contribute to' section.

Search GitHub

Explore Gist Blog Help

bramadams + ⌂ ⚙ ⌂

bramadams News Feed Pull Requests Issues

GitHub Bootcamp

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

Access to thousands of Git-based projects via GitHub

(⌚) 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 v3 documentation page. At the top, there's a navigation bar with links for Docs, Blog, Forum, Versions, and a search bar. Below the header, the page title is "REST API v3". On the left, there's a sidebar with a yellow box containing the question "How can we obtain GitHub data?". The main content area has tabs for Reference, Guides, and Libraries, with "Reference" selected. The main content starts with an "Overview" section, followed by a list of numbered items from i to x. To the right of the main content is a sidebar with a tree-like navigation structure.

GitHub Developer

Docs ▾ Blog Forum Versions ▾ Search...

REST API v3

Reference Guides Libraries

Overview

This describes the resources that make up the official GitHub REST API v3. If you have any problems or requests, please contact [GitHub Support](#) or [GitHub Premium Support](#).

- i. [Current version](#)
- ii. [Schema](#)
- iii. [Authentication](#)
- iv. [Parameters](#)
- v. [Root endpoint](#)
- vi. [GraphQL global node IDs](#)
- vii. [Client errors](#)
- viii. [HTTP redirects](#)
- ix. [HTTP verbs](#)
- x. [Hypermedia](#)

- ▼ Overview
- [Media Types](#)
- [OAuth Authorizations API](#)
- [Other Authentication Methods](#)
- [Troubleshooting](#)
- [API Previews](#)
- [Versions](#)
- ▶ Activity
- ▶ Checks
- ▶ Gists
- ▶ Git Data
- ▶ GitHub Actions
- ▶ GitHub Apps
- [GitHub Marketplace](#)
- ▶ Interactions
- ▶ Issues

How can we obtain GitHub data?

How can we mine GitHub Data

The screenshot shows the GitHub REST API v3 documentation page. At the top, there's a navigation bar with links for Docs, Blog, Forum, Versions, and a search bar. Below the header, the page title is "REST API v3". On the left, there's a sidebar titled "Overview" with links to various API components. A large red box on the right contains a warning message about rate-limit issues.

GitHub Developer

Docs ▾ Blog Forum Versions ▾ Search...

REST API v3

Reference Guides Libraries

Overview

This describes the resources that make up the official GitHub REST API v3. If you have any problems or requests, please contact [GitHub Support](#) or [GitHub Premium Support](#).

- i. [Current version](#)
- ii. [Schema](#)
- iii. [Authentication](#)
- iv. [Parameters](#)
- v. [Root endpoint](#)
- vi. [GraphQL global node IDs](#)
- vii. [Client errors](#)
- viii. [HTTP redirects](#)
- ix. [HTTP verbs](#)
- x. [Hypermedia](#)

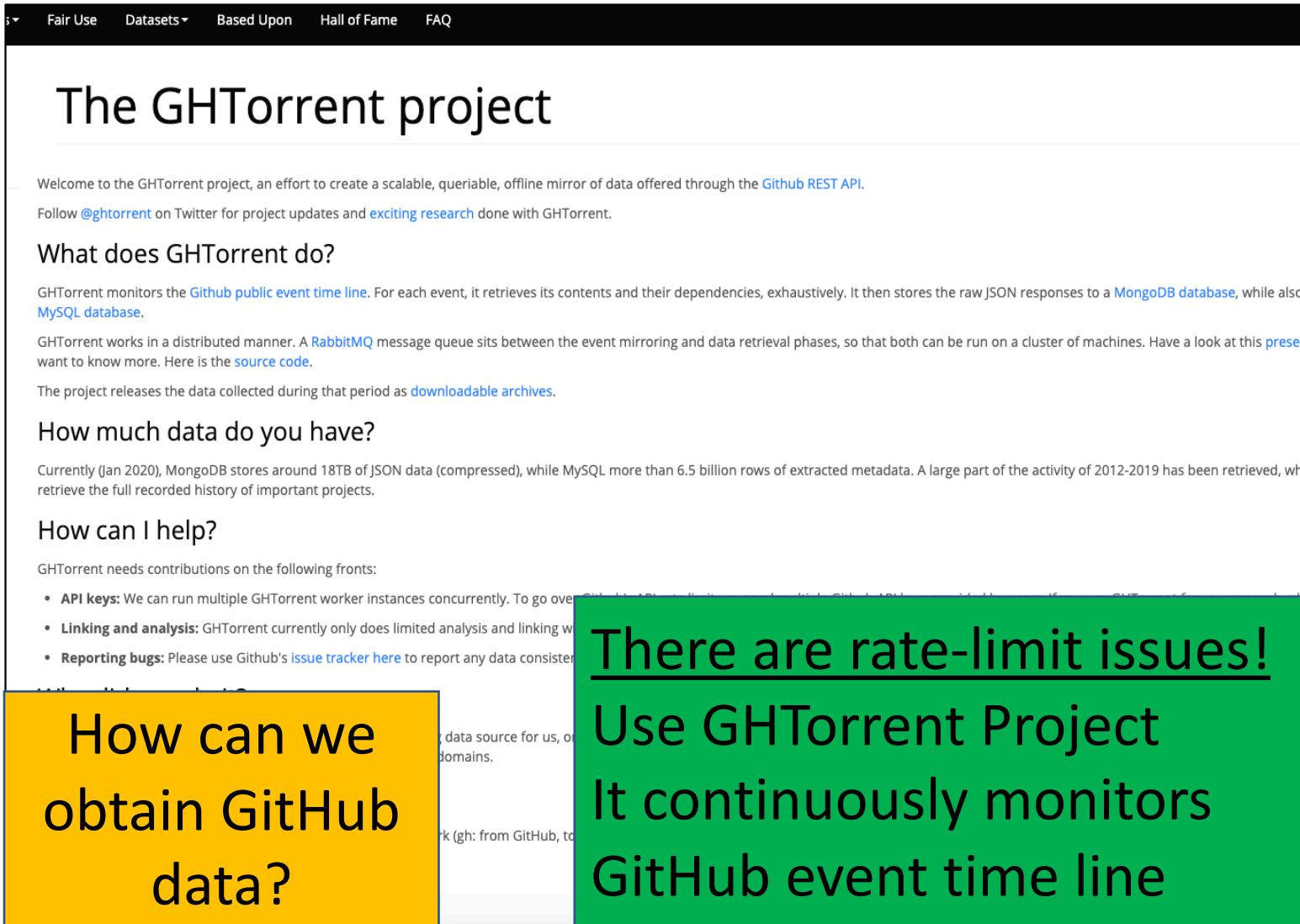
How can we obtain GitHub data?

There are rate-limit issues!

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

- ▼ Overview
- Media Types
- OAuth Authorizations API
- Other Authentication Methods
- Troubleshooting
- API Previews
- Versions
- ▶ Activity

How can we mine GitHub Data



The screenshot shows the homepage of the GHTorrent project. The header includes links for Fair Use, Datasets, Based Upon, Hall of Fame, and FAQ. The main title is "The GHTorrent project". Below it, a welcome message states: "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." A section titled "What does GHTorrent do?" explains the project's purpose: monitoring the Github public event time line, retrieving contents and dependencies, and storing raw JSON responses to a MongoDB database and MySQL database. It also mentions RabbitMQ, a distributed manner, and source code. Another section, "How much data do you have?", notes around 18TB of compressed JSON data and 6.5 billion rows of metadata. A "How can I help?" section lists API keys, linking and analysis, and reporting bugs. A yellow sidebar on the left asks, "How can we obtain GitHub data?". A green sidebar on the right highlights "There are rate-limit issues!" and "Use GHTorrent Project".

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 the rate limit, we need API keys for multiple GitHub accounts. If you have one or more GitHub accounts, you can contribute them to us.
- **Linking and analysis:** GHTorrent currently only does limited analysis and linking with other data sources. If you have domain knowledge about GitHub, please contribute your expertise.
- **Reporting bugs:** Please use Github's [issue tracker here](#) to report any data consistency issues or bugs you find.

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

The screenshot shows the GitHub REST API v3 documentation page. At the top, there's a navigation bar with links for Docs, Blog, Forum, Versions, and a search bar. Below the header, the page title is "REST API v3". On the left, there's a sidebar with a list of topics: i. Current version, ii. Schema, iii. Authentication, iv. Parameters, v. Root endpoint, vi. GraphQL global node IDs, vii. Client errors, viii. HTTP redirects, ix. HTTP verbs, and x. Hypermedia. To the right of the sidebar, the main content area has tabs for Reference, Guides, and Libraries, with "Reference" selected. The main content starts with an "Overview" section, which is currently expanded. Other sections listed under "Overview" include Media Types, OAuth Authorizations API, Other Authentication Methods, Troubleshooting, API Previews, and Versions. Below the "Overview" section, there's a green box containing the text: "There are rate-limit issues! GitHub allows one to use authentication where u get a token (5000 requests/hr)".

GitHub Developer

Docs ▾ Blog Forum Versions ▾ Search...

REST API v3

Reference Guides Libraries

Overview

This describes the resources that make up the official GitHub REST API v3. If you have any problems or requests, please contact [GitHub Support](#) or [GitHub Premium Support](#).

- i. [Current version](#)
- ii. [Schema](#)
- iii. [Authentication](#)
- iv. [Parameters](#)
- v. [Root endpoint](#)
- vi. [GraphQL global node IDs](#)
- vii. [Client errors](#)
- viii. [HTTP redirects](#)
- ix. [HTTP verbs](#)
- x. [Hypermedia](#)

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

How can we obtain GitHub data?

Some of my work

Patch Propagation in Related Applications



Universiteit
Antwerpen

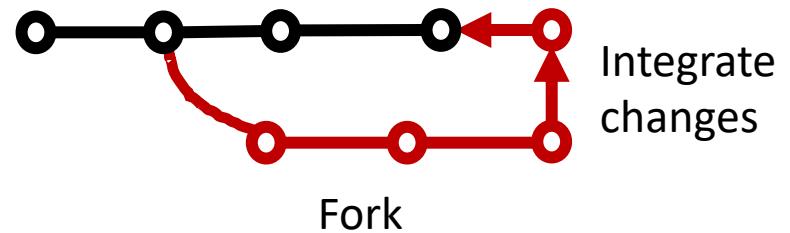
John Businge

John.businge@uantwerpen.be

Introduction



Bitbucket



Reuse in Android ecosystem

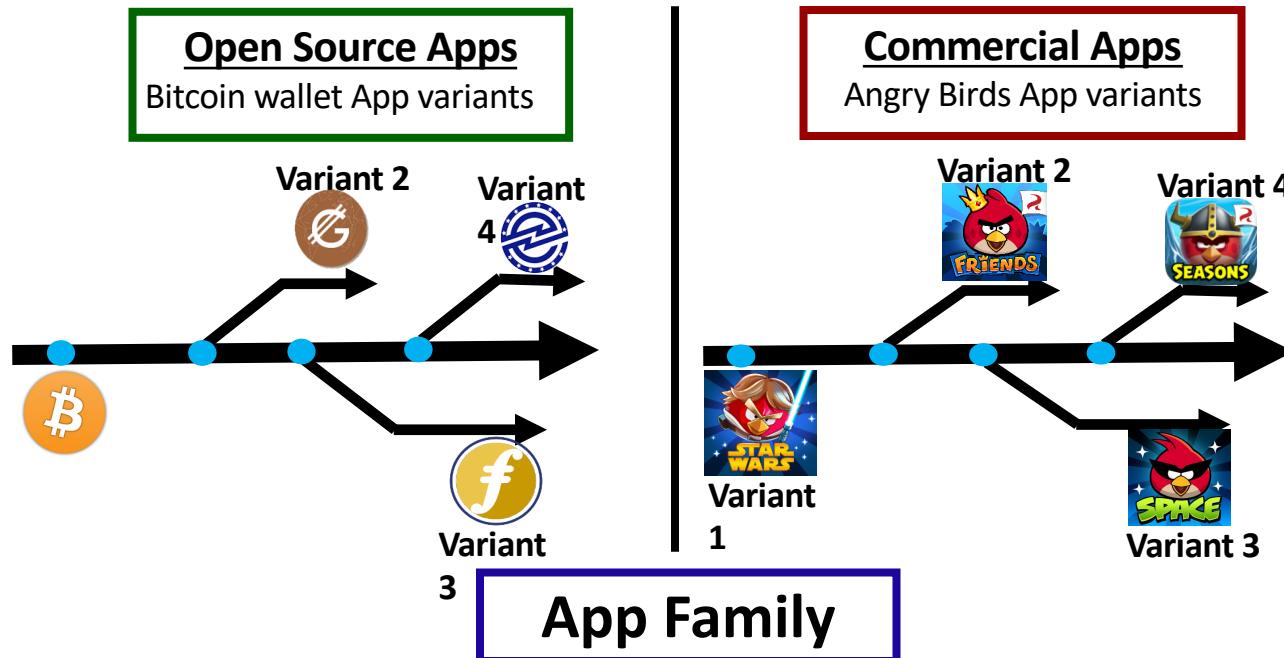


Nov - 2019
>940K repositories,
48K users



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

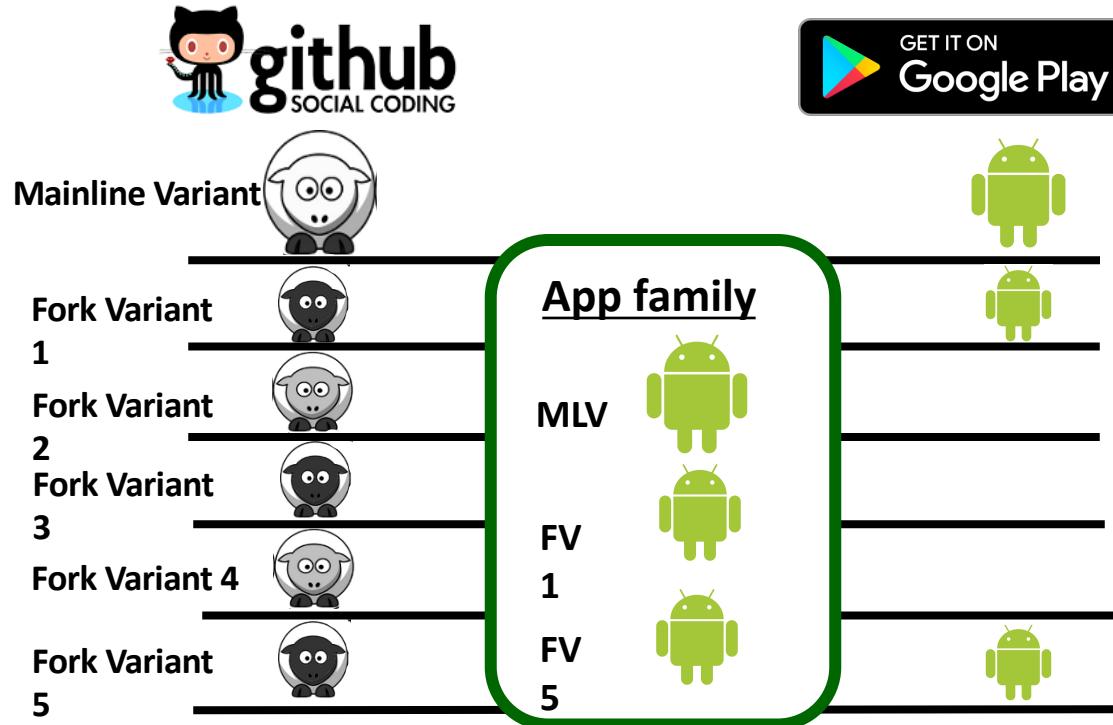
Android Mobile-app variants



Reasons for creating App Variants:

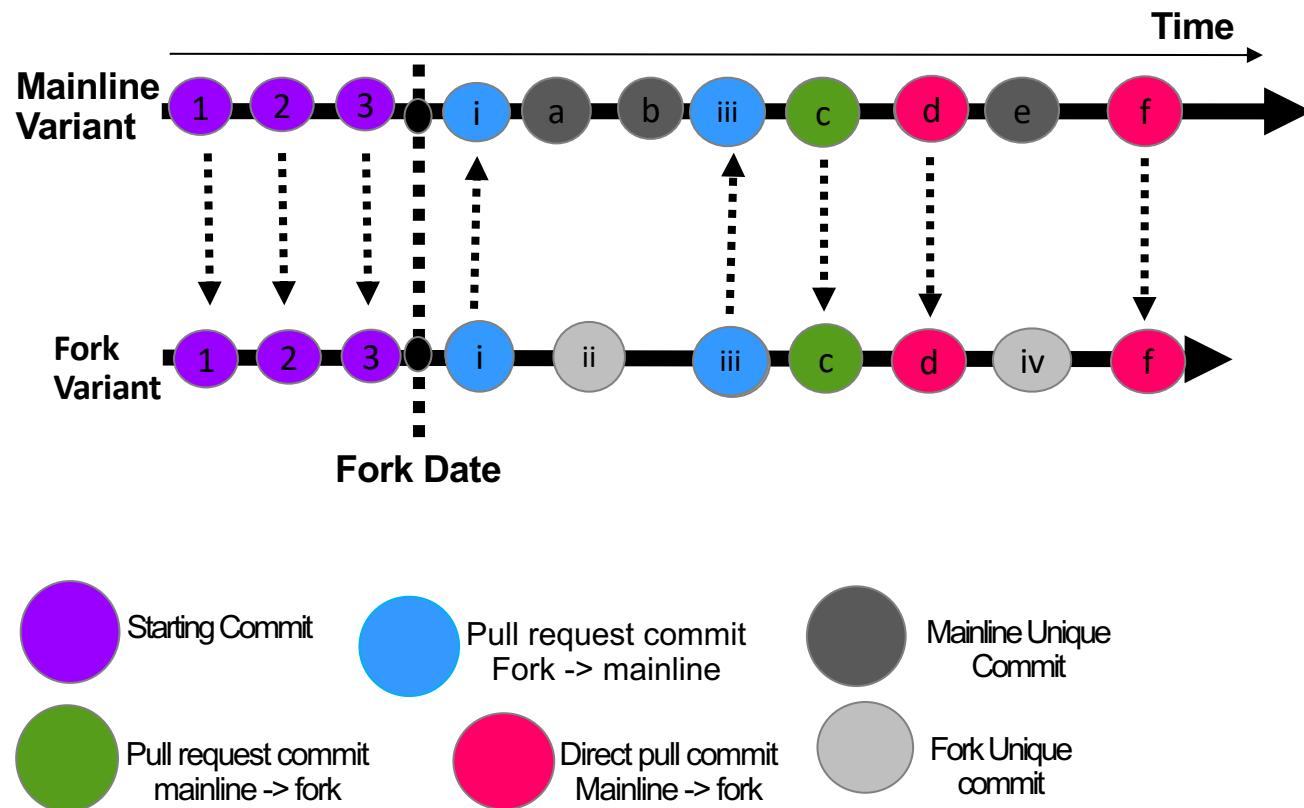
- Different user needs
- Different markets
- Different payment modes

App Family Identification

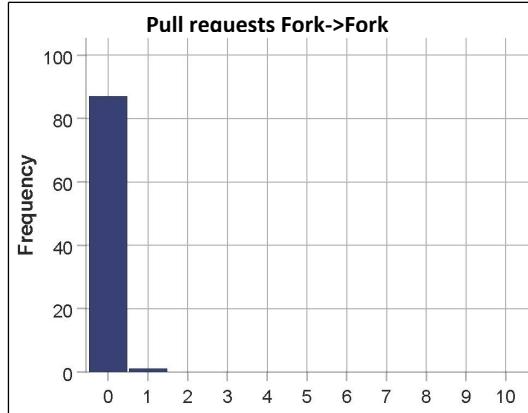
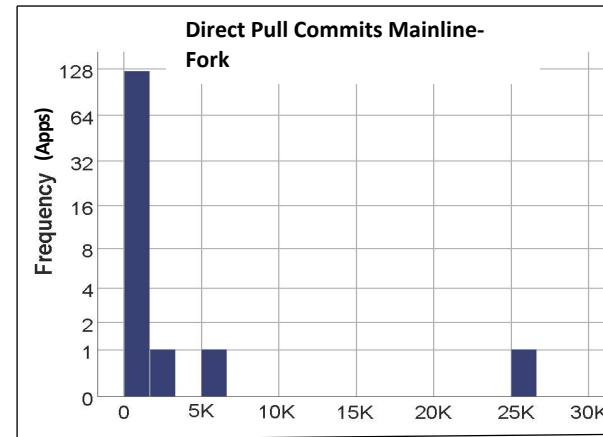
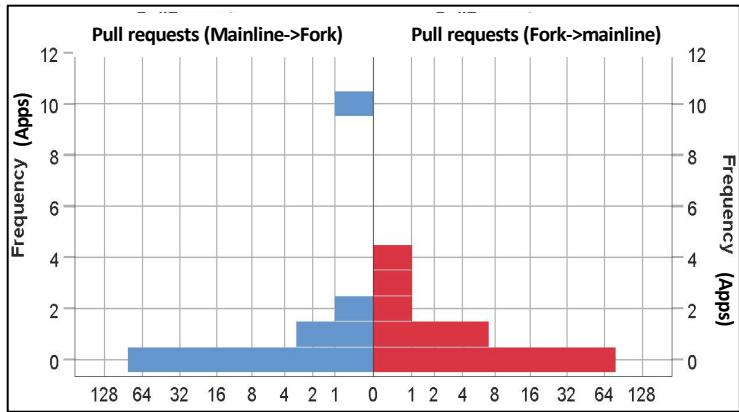


J. Businge, et al., "Clone based variability management in the android ecosystem", ICSME, 2018.

Code propagation in the Android Family



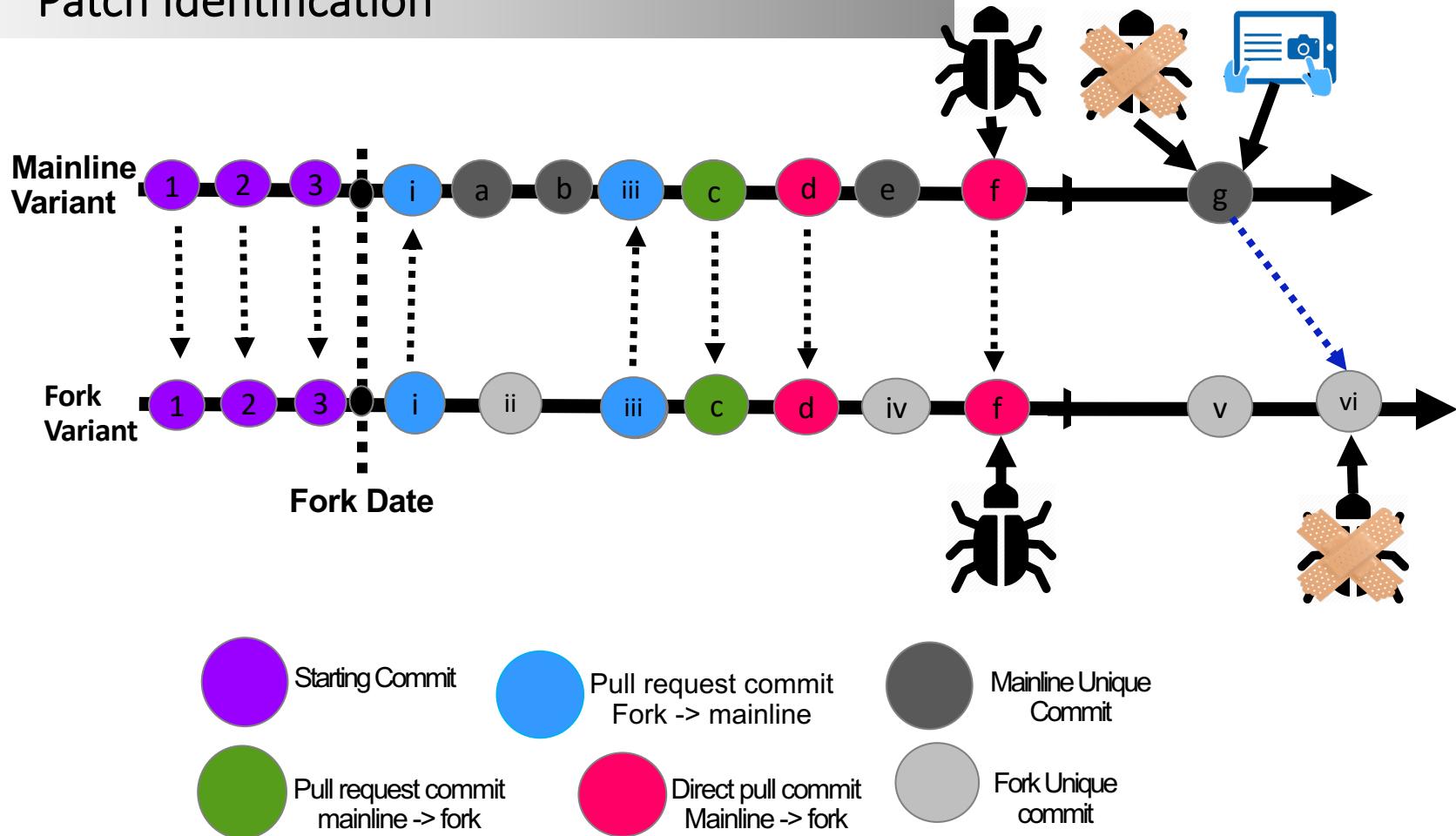
Code Propagation Results – ICSME 2018



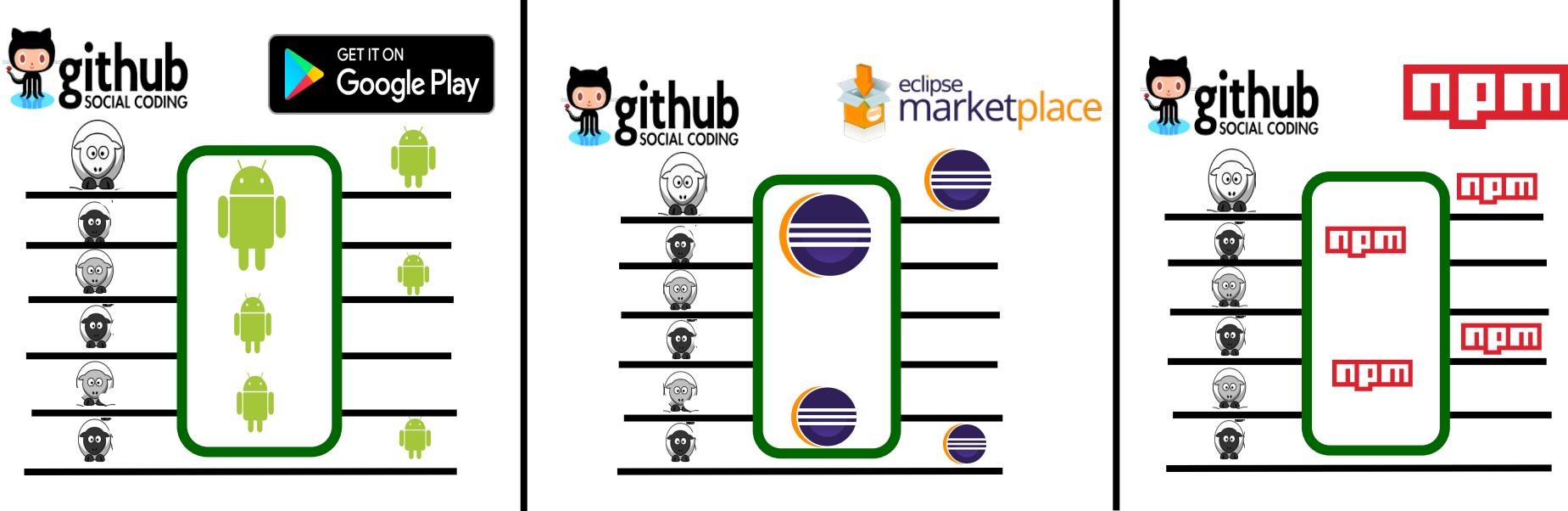
- Limited code propagation between variants
- Developer's difficulty of finding patches in shared artifacts
- GitHub only provides how many commits a variant is ahead/behind w.r.t another

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

Patch Identification



Method 1 - Collect Study Subjects

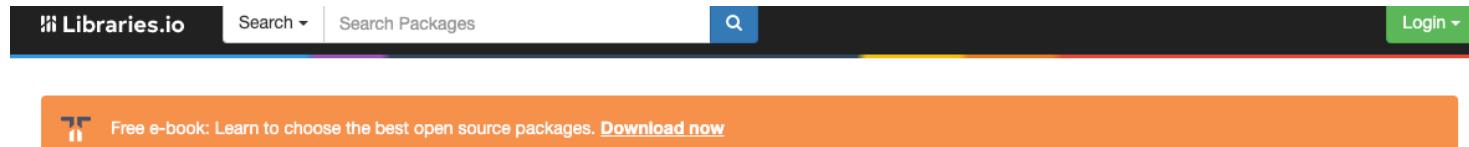


How can we mine SE Data – Part II

Package managers - Libraries.io

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

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 or [open an issue](#).

Let's get our hands dirty 😊