

Analysing Software Evolution with Diggit

SoftwareEngineering@LaBRI



Jean-Rémy Falleri



Floréal Morandat



Matthieu Foucault

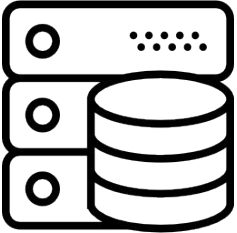
Mining Software Repository

- Use open source software repository to uncover knowledge on **software evolution**
- Leverage on **source code** and **repository metadata**
- Necessitate to use **advanced statistics** in order to validate assumptions and hypotheses

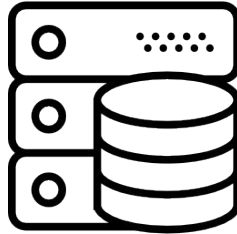
Challenge

*Do developers write tests before
code?*

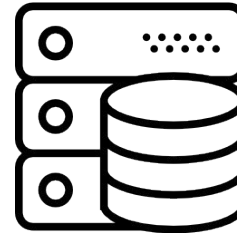
jQuery



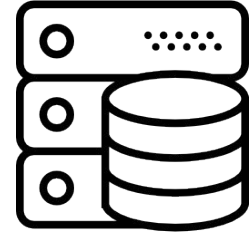
Apache



...



Eclipse

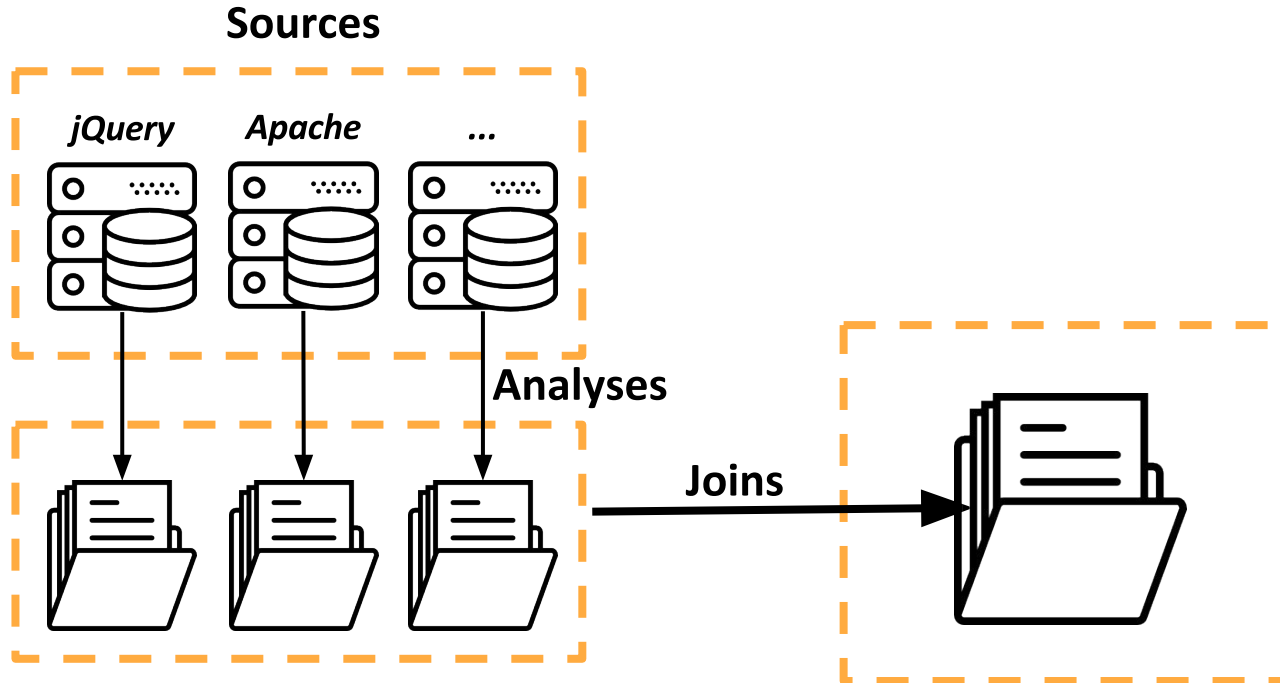


Problems


- **Heterogeneous** sources of data
- Need **various scripts** in **various languages** in order to run
- Need to use **scaling** storage technologies
- Must be easy to reuse for **reproducibility**

A handmade solution is not going to cut it...

Diggit



Diggit

- Manages a set of **sources**
- Manages a set of **analyses** that produce data out of a source
- Manages a set of **joins** that produce data out of the analyses' data
- Manages **addons** that are services to help coding analyses or joins (i.e. db addon)
- Is written in **ruby** 

Managing sources

```
$ dgit init
```

```
$ dgit sources add https://github.com/jrfaller/test-git
```

```
$ dgit sources add https://github.com/jrfaller/diggit
```

```
$ dgit sources
```

```
$ dgit clones perform
```

```
$ dgit status
```

An analysis

```
class TestAnalysis < Diggit::Analysis
  def run
    puts "Run on #{source.url}"
    puts "I have bindings to the
      repository: #{repo}"
  end

  def clean
    puts "Clean on #{source.url}"
  end
end
```

see Rugged::Repository

in plugins/analysis/test_analysis.rb

Managing analyses

```
$ dgit analyses add test_analysis
```

```
$ dgit analyses perform
```

```
> "Run on https://github.com/jrfaller/test-git"
```

```
> "Run on https://github.com/jrfaller/diggit"
```

```
$ dgit analyses perform -m clean
```

```
> "Clean on https://github.com/jrfaller/test-git"
```

```
> "Clean on https://github.com/jrfaller/diggit"
```

A join

```
class TestJoin < Diggit::Join
  require_analyses "test_analysis"
  def run
    puts "Run on #{sources}"
  end

  def clean
    puts "Clean on #{sources}"
  end
end
```

in plugins/join/test_join.rb

Managing joins

```
$ dgit joins add test_join
```

```
$ dgit joins perform
```

```
> “Run on (https://github.com/jrfaller/test-git,  
https://github.com/jrfaller/diggit)”
```

```
$ dgit joins perform -m clean
```

```
> “Clean on  
(https://github.com/jrfaller/test-git, https://github.com/jrfaller/diggit)”
```

Using addons

```
class TestAnalysis < Diggit::Analysis
  require_addons "out"
  def run
    FileUtils.mkdir_p(out.out)
  end

  def clean
    out.clean
  end
end
```

Exercises

- Display on the standard output all commits from the repository <https://github.com/jrfaller/test-git>
- Display all commit messages
- Display all committers