경험적 연구를 위한
GitHub 오픈소스 소프트웨어
데이터 수집

KCSE 2021 튜토리얼

서울과학기술대학교
김진대

Objectives

- 경험적 연구를 위한 데이터 수집은 연구 진행에 있어 매우 중요한 부분이지만 논문에서는 간단히 설명되는 경우가 많음
    - 하지만 경험이 쌓이기 전까지는 무턱대고 손대기 쉽지 않음.
    - 좋은 아이디어가 있고 궁금한 점이 있지만 어떻게 데이터를 모아 확인해야 할 지 모르겠는 경우.
- GitHub에서 데이터 수집하며 겪었던 경험 공유
    - 데이터 수집 계획의 참고자료로 활용, 유사한 시행 착오를 피하여 데이터 수집 시간을 단축.

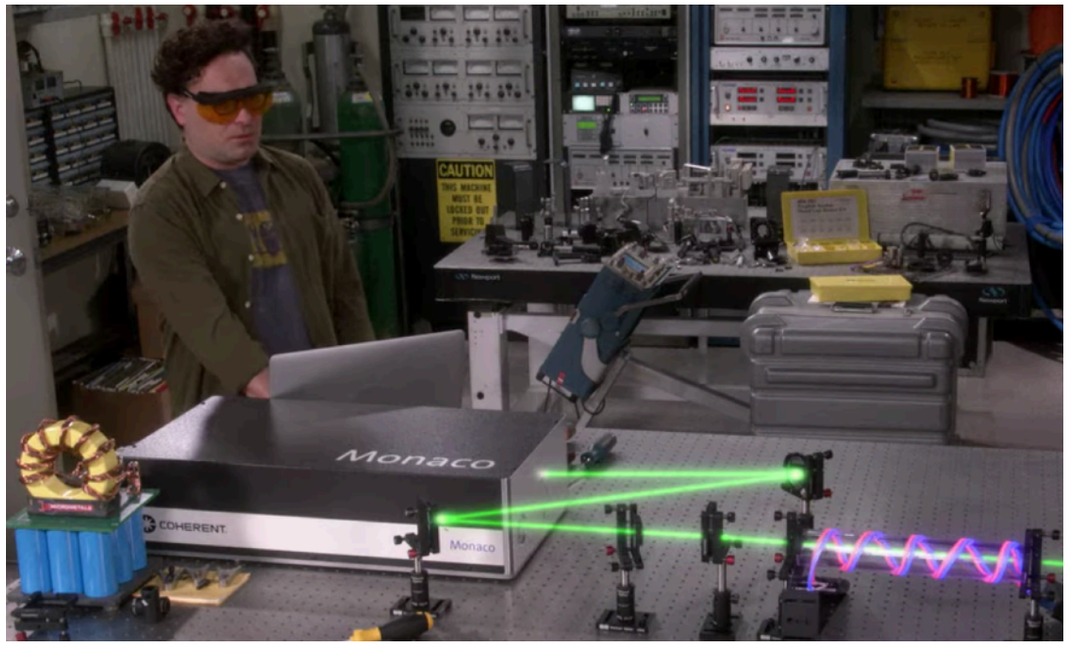
경험적 연구
Empirical Study

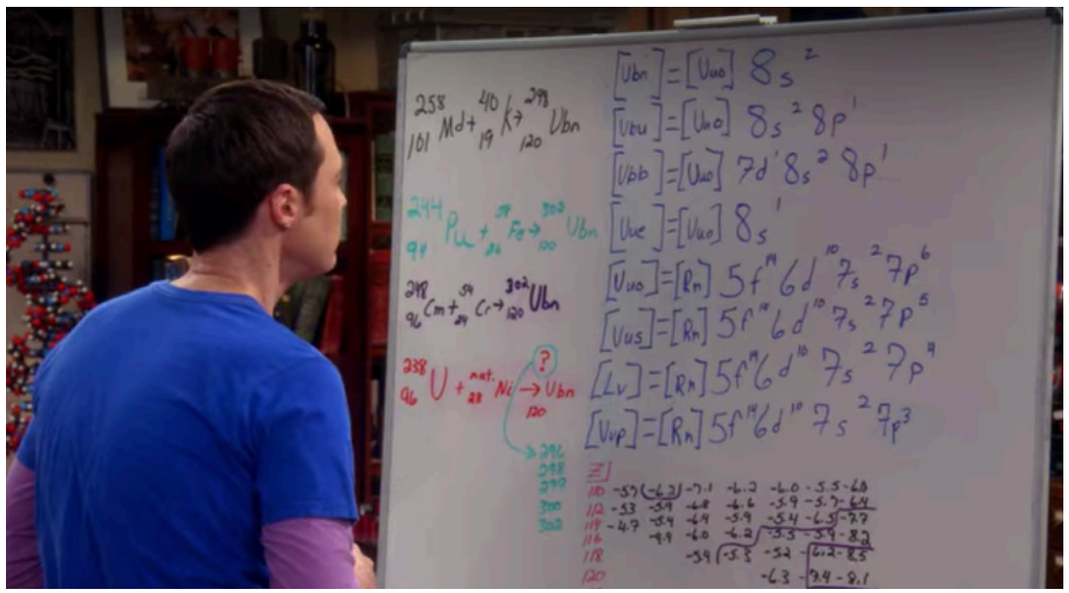
- 관찰이나 실험 등을 통해 새로운 이론이나 지식을 얻고자 하는 연구방식.

- 실증적인 연구라고도 합니다.

- 개념적(Conceptual) 연구와 대비됩니다.

- e.g.) Leonard vs. Sheldon in Big Bang Theory





경험적 연구 in SE

- 많은 연구들이 경험적 연구 부분을 포함하고 있음.
- 실제 개발과정에서 어떤 일이 일어나고 있는지 흥미로운 사실을 밝혀내기 위한 연구.
- 새롭게 개발한 기법이 정말로 도움이 될 것인지를 수집된 실제 개발 데이터를 이용하여 검증.
- 오픈소스 소프트웨어 프로젝트에서 수집된 데이터를 많이 활용.

데이터 수집단계의 중요성

- 잘못된 데이터 → 잘못된 결론
- 질문에 대한 답을 찾기 위해 데이터를 분석하는 경우,
  - 분석한 데이터 자체가 잘못되었다면 분석이 의미 없음.
- 새로 개발한 기법을 검증/평가하기 위해 수집된 데이터 사용
  - 내가 수집한 데이터에서만 결과가 잘 나오는 것은 아닌가?
  - 수집한 데이터에 문제가 있어 결과가 안 나오는 것은 아닌가?

GitHub

- 수많은 소프트웨어가 개발, 관리되고 있는 플랫폼.
- 1억개 이상의 저장소(Repository), 5,600만 이상의 개발자, 3백만 이상의 조직에 대한 정보를 포함하고 있는 데이터의 원천.[1]
- 과거 SourceForge 등에서 힘들게 모아야 했던 오픈소스 소프트웨어 데이터를 수십만~수백만 규모로 보다 손쉽게 수집할 수 있음.
- 다수의 연구들이 GitHub에서 데이터를 수집하여 수행되었고, 이런 데이터를 수집하여 공유하는 프로젝트들도 진행되었습니다.

1. https://github.com 메인 페이지 정보

Repository vs. Project

- 과거 하나의 프로젝트는 보통 하나의 저장소에서 관리되는 경우가 많았음.

- GitHub에서는 하나의 프로젝트 = 하나의 저장소가 꼭 성립하지는 않음[1].

  - 저장소를 Clone하여 새롭게 저장소를 만들고 독립적인 변경사항을 추가할 수 있음.

  - 하나의 프로젝트가 여러 저장소에 분산되는 일이 일어나고, 프로젝트 전체에 대한 정보를 얻기 위해서는 이런 저장소들을 모두 확인해야 함.

[1] Kalliamvakou, Eirini, et al. "An in-depth study of the promises and perils of mining GitHub." *Empirical Software Engineering* 21.5 (2016).

Fork

- 한 저장소에서 분기를 만들어 새로운 저장소를 생성하는 것.
- 어떤 저장소가 충분히 의미있는 데이터를 담고 있는지 구분하는 지표로 사용 가능.
  - 어떤 저장소가 Fork되었다.
    - 그 저장소의 내용에 관심을 가지고 복사한 사람이 있다.
  - 어떤 저장소가 다른 저장소로부터 Fork된 저장소이다.
    - 생성된 이후 충분한 활동이 없다면 단지 다른 저장소의 복제에 그칠 확률이 높음.

Pull Request

- 한 저장소를 Clone하여 생성된 저장소에서 만들어진 변경사항을 원래의 저장소에 합치기 위한 요청.

- Pull Request 내역으로부터 Code Review, Issue Management에 대한 정보를 얻기 좋음[1].

- GitHub에서는 Pull Request 정보를 Issue와 비슷하게 관리.

  - Issue Report + Commit 정보를 연결하여 얻을 수 있음.

[1] Kalliamvakou, Eirini, et al. "An in-depth study of the promises and perils of mining GitHub." *Empirical Software Engineering* 21.5 (2016).

Non-Software Repositories

648 commits | **1** branch | **0** packages | **0** releases | **135** contributors

Branch: **master** | New pull request | Find file | Clone or download

leereilly Merge pull request #436 from GNi33/patch-1 ... 1 Latest commit e36c52f on 22 Aug 2018

README.md — Merge pull request #465 from Aristarhys/master — 2 years ago

README.md

Games on GitHub

Below is a list of open source games and game-related projects that can be found on GitHub - old school text adventures, educational games, 8-bit platform games, browser-based games, indie games, GameJam projects, add-ons/maps/hacks/plugins for commercial games, libraries, frameworks, engines, you name it.

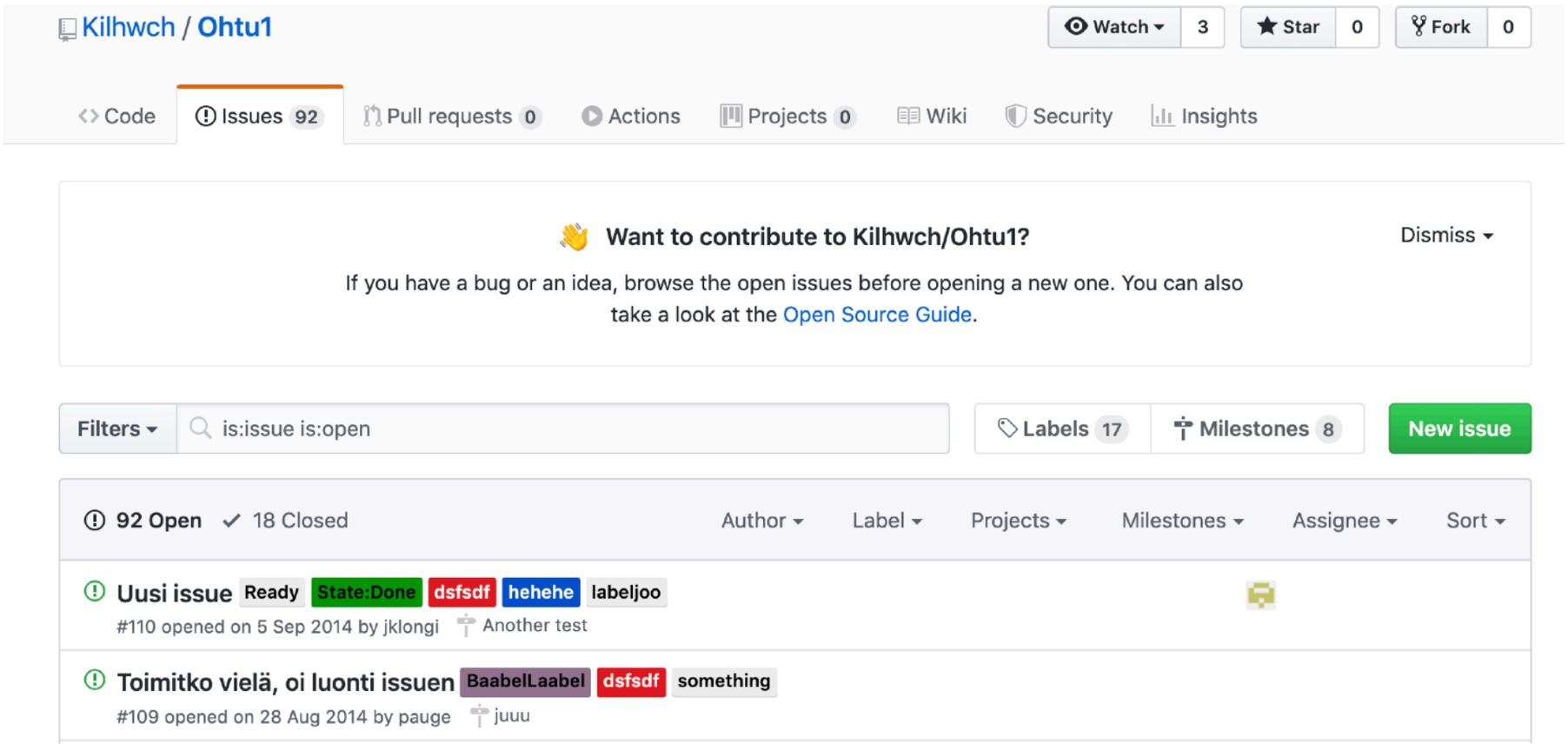
Contributing

If you'd like to add a repository to the list, please create an Issue, or fork this repository and submit a pull request (click here to edit this file from github).

Would you like to help maintain and improve this repository? Click here for information on becoming a maintainer.

Help: MarkDown Help, Markdown Cheatsheet

Non-Software Repositories



- 상당히 많은 Repository가 소프트웨어 개발과는 상관없는 데이터를 갖고 있음
- 단순한 기준으로 걸러내기 쉽지 않음
- Fork, Star, Language 등의 정보를 복합적으로 활용

Issue Management

- 코드 관리와 통합된 이슈 트래커(Issue Tracker)로 이슈들을 관리.

- 코드와 이슈를 직접적으로 연결해 주지는 않음.

- Markdown을 이용하여 이슈 내용에서 쉽게 특정 commit이나 관련된 사람들, 또 다른 이슈 등을 참조할 수 있음.

Issue Labels

- 이슈 리포트의 다양한 정보들을 레이블(Label)로 표시
- 레이블은 각 프로젝트별로 마음대로 지정하여 사용 가능
- 기존의 다양한 항목들이 모두 합쳐져 있음

Commons Collections / COLLECTIONS-756

CollectionUtils.filter(..) broken with blank Collections.singleton

Details

Type:	Bug	Status:	CLOSED
Priority:	Major	Resolution:	Not A Problem
Affects Version/s:	4.4	Fix Version/s:	None
Component/s:	Collection		
Labels:	None		

Description

When using `CollectionUtils.filter(..)`, if the passed collection is initialized with jdk `Collections.singleton(..)` and only contains an `isBlank(..)` string, the method will throw an exception instead of removing blank item as expected. Tested with commons-collections:4.4 and java8.

All can be resumed with

```
CollectionUtils.filter(Collections.singleton(" "), StringUtils::isNotBlank);
```

Bug 250991 - Reorganize options for prefix

Status:	NEW
Alias:	None
Product:	JDT
Component:	Core (show other bugs)
Version:	3.5
Hardware:	PC Windows XP
Importance:	P3 enhancement (vote)
Target Milestone:	---
Assignee:	David Audel ECA
QA Contact:	

Data Source

- **미리 수집된 데이터셋 사용**
    - GHTorrent, GHArchive
- **직접 수집하기**
    - GitHub REST API

GHTorrent

- GitHub에서 제공하는 public data를 수집하여 mysql에 저장.
  - 월별로 축적된 data dump를 제공 - download 또는 Google BigQuery 이용.
  - 저장소, 사용자, 이슈, 이벤트 등의 정보를 관계형 데이터베이스에 저장하여 분석에 편리.
- 데이터는 다음의 주소에서 다운로드 받을 수 있습니다.
  - http://ghtorrent-downloads.ewi.tudelft.nl/
  - 가장 최신 데이터는 2020-07 데이터로, 크기는 압축하여 약 130 GB.
- 데이터에 관련된 자세한 내용은 MSR'12에서 발표된 다음 논문을 참조하세요.
  - Gousios, Georgios, and Diomidis Spinellis. "GHTorrent: GitHub's data from a firehose." MSR'12

GHArchive

- http://www.gharchive.org
- GitHub의 public event 데이터를 저장하여 분석할 수 있도록 제공.
- GitHub에서 제공하는 JSON 형태의 event가 그대로 저장되어 있음.
- Google BigQuery에서 SQL을 사용하여 보다 복잡한 질의를 하는 것도 가능.

GitHub REST API

- RESTful API를 사용하여 다양한 정보를 요청하고 받아 올 수 있음.
- 최신 데이터를 수집할 수 있는 유일한 방법.
    - 최신 데이터에만 접근할 수 있다는 것이 단점.
    - e.g.) event 데이터의 경우 최근 90일, 최대 300개의 event만 제공됨.
- GitHub REST API 공식 문서
    - https://docs.github.com/en/rest

GHTorrent 활용하기

- Data dump를 받아 압축을 풀고 MySQL 서버에 import하여 사용.

- Import 하기위한 스크립트를 적절히 수정하면 필요한 테이블만 선택하여 저장할 수 있음.

- 테이블 구조를 참조, SQL을 이용한 질의(Query)로 보다 복잡한 분석을 진행할 수 있음.

  - e.g.) 저장소별 평균적으로 이슈가 해결되는데 걸리는 시간.

GHTorrent의 문제점

- 가장 큰 문제점은 현 시점의 최신 데이터가 2020년 7월까지의 데이터로 반 년에 가까운 차이가 생긴다는 점.
    - 많은 데이터가 현재의 GitHub과는 일치하지 않게 됨.
    - 저장소가 다른 사용자/이름으로 migration되는 경우도 자주 발생.
- 저장된 데이터 안에서도 수집과정에서의 문제 등으로 데이터 일관성/무결성 등이 제대로 보장되지 않음.
    - e.g.) 이슈의 생성 날짜(created_at)이 종료 날짜(closed_at)보다 더 나중인 경우, event로 저장된 내역과 실제 테이블에 저장된 정보가 일치하지 않는 경우.
    - 많은 경우 분석하기 전 데이터의 검증이 꼭 필요합니다.

GitHub REST API로 보완하기

- GHTorrent의 정보를 그대로 사용하기 보다는 데이터를 수집할 대상을 추리는데 사용.

- 대상이 되는 저장소의 최신 정보를 GitHub REST API를 이용하여 수집하는 것이 더 적절할 수 있음.

- e.g.) GHTorrent에서 생성된 지 최소 3년이 넘었고, Pull Request가 100개 이상인 저장소의 목록을 뽑아냄

    - 이 저장소들에 대해서 GitHub REST API를 이용하여 최신의 정보까지 모두 수집.

    - Event 데이터와 같이 과거 내역이 필요한 경우에는 GHTorrent/GHArchive 등의 데이터를 활용.

GitHub REST API로 수집하기

1. 연구를 위해 어떤 정보를 수집해야 하는지 확인.

2. GitHub REST API 문서에서 요청과 응답의 형식을 분석

3. 원하는 데이터를 얻기 위해 필요한 URL 만들기

4. 요청하여 얻어 온 응답(JSON 형식)에서 원하는 데이터만 골라 적절한 형식으로 저장.

    - 여러 번의 요청 사이 동일한 데이터 식별은 GitHub이 부여한 ID를 사용하면 됩니다.

Crawling Tips

- https://api.github.com/xxx로 요청
    - GitHub의 웹페이지에 보이는 정보들은 대부분 가져올 수 있다고 생각하시면 됩니다.
- 소프트웨어 관련 저장소를 수집하기 위해 저장소 정보의 languages 항목을 참조.
    - 저장소에 C, Java, Python 등을 포함한 경우에만 데이터를 수집하는 것도 가능.
- 저장소가 삭제된 경우 error 응답이 오거나, migration된 경우 요청이 자동으로 redirect 되는 경우도 있음.
- 사용자 입력이 가능한 string 데이터에는 이모티콘 사용이 매우 빈번하므로 encoding에 주의 → 4바이트를 지원하는 UTF-8 사용, e.g.) utf8mb4 in MySQL

Rate Limit Issue

- 하나의 URL을 사용하여 데이터를 읽어오는 것은 1개의 요청으로 간주됨.

- GitHub에서는 이런 요청을 사용자 1명당 1시간에 5,000번으로 제한하고 있음.

- 100개의 프로젝트에서 평균 5천개의 이슈 수집 → 100시간 ≒ 4일

- 보다 대량의 데이터를 수집하려고 한다면?

Rate Limit Tips

- 개별 데이터보다는 전체 데이터를 수집하여 요청횟수를 줄임.
    - 일반적으로 최대 100개의 데이터를 1페이지에 표시(일부는 100개보다 적기도 함).
    - 각각의 이슈를 따로 요청하기보다는 이슈들이 속한 저장소의 모든 이슈를 가져와서 분류하는 것이 나을 수 있음.
    - 이슈에 대한 데이터를 요청 → 이슈와 관련된 사용자 + 레이블 정보도 응답에 포함되어 있음.
- 수집을 위해 여러 사용자를 등록하여 사용.
    - 이메일 주소만 있으면 여러 명의 사용자를 등록하여 Rate Limit을 늘릴 수 있음.

GitHub Crawler

- GitHub REST API에서 데이터를 수집할 수 있는 도구
  - https://github.com/Jindae/github_crawler
    - 설정에 등록된 사용자들의 Rate Limit을 번갈아가며 사용.
    - 지정된 URL의 모든 페이지 수집.
    - 일부 예외처리 - 유효하지 않은 URL 등.
- 예제로 파일로 주어진 저장소별로 이슈 정보들을 수집하여 MongoDB에 저장하도록 Python으로 구현되어 있음.

PAT 발급 받기

- Personal Access Token (PAT)은 Crawler로 사용자 인증을 받기 위해 꼭 발급 받아야 합니다.

- 다음 주소에서 자세한 설명을 볼 수 있습니다.
  - https://docs.github.com/en/github/authenticating-to-github/creating-a-personal-access-token

- 발급 받은 후 Crawler의 settings.json에 등록해 주면 됩니다.

  - 인증이 없다면 1시간에 단 60번의 요청만 할 수 있습니다.

설정 및 실행

- 테스트 해 보기 위해서는 settings.json 파일에 발급 받은 PAT 정보를 입력하고, MongoDB 접속 정보를 설정해 주면 됩니다.

- requests, pymongo를 사용하므로 미리 설치해 주세요.
    - pip install requests

- github_crawler.py를 실행.
    - 등록된 사용자의 Rate Limit 정보는 rate_limit_info.py로 확인 가능.

원하는대로 고쳐쓰기

- 다른 정보를 수집하고 싶다면?
  - Crawler를 Fork해서 마음대로 수정하여 사용하면 됩니다.
  - 요청을 위한 URL을 생성하는 부분과 받아 온 정보를 분석하여 저장하는 부분을 적절히 수정하세요.
  - GitHub REST API 문서를 참조하면 원하는 정보를 얻기 위한 URL, parameters, response의 포맷 등을 확인할 수 있습니다.

요약 및 정리

- SE의 경험적 연구를 위해 소프트웨어 개발 관련 데이터를 수집하는 것이 매우 중요함.

- GitHub은 이런 데이터를 풍부하게 포함하고 있고, 연구에 자주 사용되고 있습니다.

- 단, 유용하고 믿을 수 있는 데이터를 얻기 위해서는 그만큼 고민과 시간을 투자해야 합니다.

감사합니다