A Versatile Dataset of Agile Open Source Software Projects

Vali Tawosi, Afnan Al-Subaihin, Rebecca Moussa, Federica Sarro
{vali.tawosi,a.alsubaihin,rebecca.moussa.18,f.sarro}@ucl.ac.uk
University College London
London, UK

ABSTRACT
Agile software development is nowadays a widely adopted practise in both open-source and industrial software projects. Agile teams typically heavily rely on issue management tools to document new issues and keep track of outstanding ones, in addition to storing their technical details, effort estimates, assignment to developers, and more. Previous work utilised the historical information stored in issue management systems for various purposes; however, when researchers make their empirical data public, it is usually relevant solely to the study’s objective. In this paper, we present a more holistic and versatile dataset containing a wealth of information on more than 500,000 issues from 44 open-source Agile software, making it well-suited to several research avenues, and cross-analyses therein, including effort estimation, issue prioritization, issue assignment, and many more. We make this data publicly available on GitHub to facilitate ease of use, maintenance, and extensibility.

CCS CONCEPTS
• Software and its engineering;

KEYWORDS
Agile Development, Open-Source Software, Data Mining

ACM Reference Format:
Vali Tawosi, Afnan Al-Subaihin, Rebecca Moussa, Federica Sarro. 2022. A Versatile Dataset of Agile Open Source Software Projects. In Proceedings of MSR ’22: Proceedings of the 19th International Conference on Mining Software Repositories (MSR 2022). ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nmnnnn.nnmmnn

1 INTRODUCTION
The early 2000s has witnessed a surge of the adoption of Agile Software Development alongside the release of the Agile Software Development Manifesto in 2001 [12]. Agile techniques boast a faster response to unanticipated alterations that can arise during development such as changes in user requirements, development environments and delivery deadlines; typically contrasted with traditional ‘plan-based’ project development, which operates under the assumption that software is specifiable and predictable [11]. Agile Software Development is currently among the most common software development methods in project management [24].

Managing agile software development is commonly aided by an issue tracking tool, which allows agile teams to log and organize outstanding development tasks (e.g., bug fixes, functional and non-functional enhancements), in addition to hosting meta-data related to these tasks. Issue tracking tools, such as Jira [1], provide a trove of historical information regarding project evolution that promise great value for Empirical Software Engineering research. Such data has been employed to address many software engineering problems such as effort estimation [9, 28], task prioritization [13, 15, 31], task assignment [20], task description enhancement [7], iteration planning [8] and exploring social and human aspects [21, 22, 32, 33]. However, the data made available by previous empirical studies is usually mainly relevant solely to the study’s objective. Therefore, we aim at paving the way for a more holistic and versatile dataset containing a wealth of information on open-source software, projects, which can serve as a single source for many possible research avenues, and enable novel investigations on the inter-play of multiple factors as well as draw observations across multiple research studies.

We call this dataset the TAWOS (Tawosi Agile Web-based Open-Source) dataset. It encompasses data from 13 different repositories and 44 projects, with 508,963 issues contributed by 208,811 users. The dataset is publicly hosted on GitHub [29] as a relational database, and designed such that it is amenable to future expansions by the community. Prospective contributors are welcome to join our effort to maintain, grow and further enhance the database by issuing a pull request on Github.

2 DATASET DESCRIPTION
2.1 Data Extraction
This dataset was mined during the latter half of October 2020. The mining process targeted 13 major open source repositories: Apache, Appcelerator, Atlassian, DNNSoftware, Hyperledger, Lsstcorp, Lyra-sis DuraSpace, MongoDB, Moodle, MuleSoft, Spring, Sonatype, and Talendforge. Most of these repositories were employed by previous work and they all used Jira as an issue management platform, which ensures uniformity of structure and availability of information. From each of these repositories, projects were selected such that they a) employ agile software development methodologies and b) have at least 200 issues with recorded story point entries. While the motivation behind the first restriction is clear, we impose the second in order to have enough data to enable statistical analyses resulting in meaningful conclusions.

A total of 904 projects from the aforementioned repositories were considered, among which we selected the 44 that satisfy the collection constraints. To extract issue information, we used the Jira Rest Java Client (JRJC) [3]; JRJC was used alongside our own
tool, implemented in Java, to extract further features that are not implemented in JRJC (see Section 2.5).

### 2.2 Data Storage

The final dataset is modeled and stored as a relational database. This enables users of the dataset to employ SQL for easy horizontal and vertical data sampling in addition to allowing easier future expansion. We elected to host the dataset in the MySQL Database Management System as it is lightweight and ubiquitous. The database can be downloaded from a GitHub repository together with the instructions on how to install and use it [29].

### 2.3 Data Characteristics

The TAWOS dataset contains 508,963 issues from 44 project. The projects are diverse in terms of different project characteristics. Each project contains issues that range from 313 to 66,741 issues. The projects span different programming languages, different application domains and different team geographical locations. Table 1 shows the number of various elements for each of the projects contained in the dataset currently. Those include the number of: issues, Bug types, distinct users (i.e. bug report contributors, etc.), developers, change logs and comments, components, sprints, versions, and the number of issues with story points assigned.

### 2.4 Data Structure

Figure 1 shows the Entity-Relationship Diagram of the database. The core entity is the Issue table, which holds the main information about an issue report. Some of its fields are directly extracted from the issue report, whereas others are derived from the information stored or the events that occurred during the issue’s lifecycle. We elaborate on these in Section 2.5.

Other important tables are Comment and Change_Log tables. Comments hold the documented discussions of the team around the issue development. Change logs hold all the changes made by the users on the issue report, by recording the field that received the change, the previous value, the next value and the nature of the change. Both these tables store the chronological order of the events in the creation_date field. Information about the Sprints, Versions, and Components of the issues are also stored in separate tables. The Issue_Links table captures the links between the issues. The User table stores all the distinct users who interacted with each project, in addition to linking the events and information to their authors and user roles. Any personally identifiable information of users like their usernames and emails are redacted from this dataset.

### 2.5 Computed and Derived Fields

To further enrich the dataset, we have augmented the mined data with several additional features that are computed or derived from the source Jira repositories as described below.

**Issue Description Text and Code.** The Description field holds the long description of the user story or bug report which can contain natural text interleaved with code snippets or stack-traces. To facilitate processing, we separate the code snippets/stack traces and the natural text describing the issue into the Description_Code and Description_Text fields respectively. We maintain the original description in the Description field. Same is done for the Comment field, from which we extract the Comment_Code and Comment_Text. This is motivated by previous work showing that code tokens may have different meaning from those found in natural language text, hence ought to be analysed separately [23, 26, 28].

**Resolution Time.** The field Resolution_Time_Minutes stores the time span (in minutes) between when an issue is marked for the first time as “In Progress” and when it is marked as “Resolved”. This period can be considered as an approximation of the time taken by the development team to resolve the issue. This is usually the target variable used for bug resolution/fixing time estimation [14, 18, 27]. Other proxies for time are provided, such as In_Progress_Time and Total_Effort_Time, indicating, respectively, the implementation time and the development (including code review and testing) time.

**SP Estimation Date.** This field records the time when the Story_Point field of the Jira issue report was populated by the developer. This information might be useful, for example, for studies on software effort estimation, in order to properly take into account the chronological order of the estimates and avoid unrealistic usage of the data as described in previous studies [6, 17, 25].

**Date and Time.** The date and time stored in different Jira repositories may have different timezones, as the projects usually have contributors from all around the World. Therefore, we converted and stored all dates and times to a unified timezone, namely the Coordinated Universal Time (UTC).

**Field Change Flag.** It is important to keep track of the changes developers made to some of the issue fields. For example, the title and description of the issue are two important pieces of information used by recent automated approaches to produce effort estimates [9]. therefore it is important to know whether these fields have been edited after the initial estimate was done. The Title_Changed_After_Estimation and Description_Changed_After_Estimation fields store this flag. We also provide a flag that shows whether the SP has been changed after the initial estimate. Note that these flags are based on the change logs of the issue.

**Change Type in Change Log.** This field is calculated to categorise change log updates into one of five categories: “STATUS” indicates the change is of transition type, which moves an issue from one status to another in the Jira workflow; “DESCRIPTION” indicates that the change was made in the issue title or description; “PEOPLE” indicates that the assignee or reporter of the issue were changed; “STORY_POINT” indicates that the Story Point field of the issue has been updated. Any other changes were categorised as “OTHER”.

### 2.6 Extensibility and Maintainability

The TAWOS database is designed such that it is easily extensible by attaching additional information to the corpus. This can help facilitate studying different problems and/or aspects of the same problem. Sharing and managing the dataset as a GitHub repository, enables us to update, expand and enrich its content, whether by us or by the community as external contributions (i.e., pull requests). Github also guarantees that the information can be safely stored long-term, thus preventing the issues often faced in previous work where the data provided are not reachable anymore (e.g., due to use of volatile storing platform such as institutional webpages which change when researchers move to another institution).
3 ORIGINALITY AND RELEVANCE

Previous studies have extracted information from issue reports managed in Jira to build predictive models for Story Point (SP) estimation in agile software projects [9, 23, 26], however not all of them have made their data public [23, 26]. Choetkiertikul et al. [9] shared their data in a replication package [2], however, it only consists of features considered in their study (i.e., the issue key, title, description, and story point of the mined issues).

The dataset presented herein encompasses all the projects considered in previous studies [9, 23, 26] augmented with more issues and features. Furthermore, it includes 28 additional projects, which have never been used by any of these previous studies.

Our dataset has been recently used by Tawosi et al. [30] who analysed a total of 31,960 stories from 26 projects stored in TAWOS in order to replicate and extend the work by Choetkiertikul et al. [9]. This set of issues has also been used in a recent study on the effectiveness of clustering for SP estimation [28].

We believe that the TAWOS dataset can help expedite the research in the area of Agile software development effort estimation. In addition to providing a unified benchmark for such studies, it also helps circumvent the challenges faced, and the time consumed, when mining such data from the web. For example, we note that Choetkiertikul et al. [9] could not mine the same data used in the study by Porru et al. [23] likely because the repositories mined had changed during the time period between the two studies.

The use of different data in similar studies hinders the immensely useful opportunity to draw observations from across different studies performed at different times around a certain subject matter. We hope that our dataset can help the community tackle this challenge. Although our dataset has been primarily designed to aid in software engineering estimation tasks, it also includes information relevant to other software engineering research, and it is designed to be expanded by other contributors. This allows and promotes the investigation of a wider range of SE aspects as discussed in the next section.

4 RESEARCH OPPORTUNITIES

In addition to benefiting effort estimation studies, the TAWOS dataset promises value to many other areas of software engineering research, including developer productivity studies, iteration planning and task scheduling.

An important research topic in Requirement Engineering is requirement prioritization [4, 16, 31] and, especially in an agile setting, the selection of issues for the next iteration [10, 19]. The TAWOS dataset can support such studies by providing a large collection of issues, with known priorities and iterations (i.e., Sprints and Releases) coupled with various aspects providing a full-picture view of the issues, projects and assignees. Additionally, as the dataset makes historical project evolution from multiple repositories available, it enables cross-project analysis.

The TAWOS database provides information about the versioning of the software under development. This information includes the name, description, and release date of the version, and whether it is archived or released. Versions connect to issues via two relations: Affected versions, and Fix versions. The former is the version where a bug or problem was found; whereas the latter is the version where

---

1The only exception is the MuleStudio project used by Choetkiertikul et al. [9], for which we could not find the data source on-line.
2The TAWOS dataset has 485,650 more issues in total, and 46,411 more issues with Story Points compared to the one shared by Choetkiertikul et al. [9]. It also contains more issues for each of the 16 projects included in Choetkiertikul et al. [9]’s dataset.

---

Figure 1: Entity-Relationship Diagram (ERD) for the TAWOS Issues Database.
5 FINAL REMARKS

We have indicated just some of the research avenues the TAWOS dataset could be exploited for. We envision that the wealth of information provided, coupled with the ability for other researchers to participate in the growth of the dataset, will enable novel research endeavours on the inter-play among several and different aspects of open-source agile software projects. For example, if a researcher uses our dataset to analyse the corpus of issue comments with regard to developers affects (e.g., emotions, sentiments, politeness), they can extend the dataset by issuing a pull request and thereby augmenting the existing data with the results of their investigation (e.g., augment the comments written by developers with emotions such as surprise, anger, sadness and fear). This data can be re-used in subsequent research investigating the inter-play between, for example, developer emotions and productivity.

We invite potential users of the database to consult our on-line documentation [29] before use in order to understand possible limitations and select data that best fits the aim of their investigations. We plan to curate and expand our dataset by adding other projects and features, and encourage the research community to join our effort in growing and enriching it, in order to open the door for novel research avenues.

ACKNOWLEDGMENTS

The work of Vali Tawosi, Rebecca Moussa and Federica Sarro is supported by the ERC grant no. 741278.
REFERENCES

[1] [n.d.]. Jira Issue & Project Tracking Software | Atlassian. https://www.atlassian.com/software/jira

[2] [n.d.]. Source Code and data for “A Deep Learning Model for Estimating Story Points” GitHub. https://github.com/SEAnalytics/datasets/tree/master/storypoint/IEEESE2018

[3] [n.d.]. The Jira REST Java Client, Version 5.2.0. https://mvnrepository.com/artifact/com.atlassian.jira/jira-rest-java-client-app/5.2.0

[4] Rami Hasan Al-Ta’ani and Roziafawati Razali. 2016. A framework for requirements prioritisation process in an agile software development environment: empirical study. International Journal on Advanced Science, Engineering and Information Technology 6, 6 (2016), 846–856.

[5] Wisam Hatibam Abbood Abz-Zohadi, Patanamon Thongtanunam, Hoa Khanh Dam, Chakririt Tantithamthavorn, and Aditya Ghose. 2020. Workload-aware reviewer recommendation using a multi-objective search-based approach. In Proceedings of the 16th ACM International Conference on Predictive Models and Data Analytics in Software Engineering. 21–30.

[6] Abdul Ali Bangash, Hareem Sahar, Abram Hindle, and Karim Ali. 2020. On the time-based conclusion stability of cross-project defect prediction models. Empirical Software Engineering 25, 6 (2020), 5047–5083.

[7] Oscar Chaparro, Jing Lu, Fiorella Zampetti, Laura Moreno, Massimiliano Di Penta, Andrian Marcus, Gabriele Bavota, and Vincent Ng. 2017. Detecting missing information in bug descriptions. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering. 396–407.

[8] Morakot Chotkertikutl, Hoa Khanh Dam, Truyen Tran, Aditya Ghose, and John Grundy. 2017. Predicting delivery capability in iterative software development. IEEE Transactions on Software Engineering 44, 6 (2017), 551–573.

[9] Morakot Chotkertikutl, Hoa Khanh Dam, Truyen Tran, Trang Pham, Aditya Ghose, and Tim Menzies. 2019. A Deep Learning Model for Estimating Story Points. IEEE TSE 45, 7 (2019), 637–656. https://doi.org/10.1109/TSE.2018.2792473

[10] Juan J Durillo, Yuanyuan Zhang, Enrique Alba, Mark Harman, and Antonio J Nebro. 2011. A study of the bi-objective next release problem. Empirical Software Engineering 16, 1 (2011), 29–60.

[11] Tore Dybå and Torger Dingsøyr. 2008. Empirical studies of agile software development: A systematic review. Information and Software Technology 50, 9 (2008), 833–859. https://doi.org/10.1016/j.infsof.2008.01.006

[12] Martin Fowler, Jim Highsmith, et al. 2001. The agile manifesto. Software development 9, 8 (2001), 28–35.

[13] Carlos Gavidia-Calderón, Federica Sarro, Mark Harman, and Earl T. Barr. 2021. The Assessor’s Dilemma: Improving Bug Repair via Empirical Game Theory. IEEE Transactions on Software Engineering 47, 10 (2021), 2143–2161. https://doi.org/10.1109/TSE.2019.2944608

[14] Mary Habayeh, Syed Shariyar Murtaza, Andriy Miranskyy, and Ayse Basar Bener. 2017. On the use of hidden markov model to predict the time to fix bugs. IEEE Transactions on Software Engineering 44, 12 (2017), 1224–1244.

[15] Yuekai Huang, Junjie Wang, Suin Lee, Chan-Gun Lee, Ikjun Yeom, and Honguk Woo. 2020. The importance of accounting for real-world developers in JIRA. In 2020 IEEE/ACM 13th Working Conference on Mining Software Repositories. 1–10.

[16] Lalitha Isari, Shanthanu Srinivasan, and Anil K. Jain. 2016. A framework for estimating the time-based conclusion stability of cross-project defect prediction models. IEEE Transactions on Software Engineering 42, 6 (2016), 551–573.

[17] Tore Dybå and Torger Dingsøyr. 2008. Empirical studies of agile software development: A systematic review. Information and Software Technology 50, 9 (2008), 833–859. https://doi.org/10.1016/j.infsof.2008.01.006

[18] Martin Fowler, Jim Highsmith, et al. 2001. The agile manifesto. Software development 9, 8 (2001), 28–35.

[19] Matthieu Jimenez, Renaud Rwemalika, Mike Papadakis, Federica Sarro, Yves Le Traon, and Mark Harman. 2019. The importance of accounting for real-world labelling when predicting software vulnerabilities. In Proc. of ESEC/FSE’19. 775–785.

[20] Youngseok Lee, Suin Lee, Chan-Gun Lee, Ikjun Yeom, and Honguk Woo. 2020. Continual prediction of bug-fix time using deep learning-based activity stream embedding. IEEE Access 8 (2020), 10503–10515.

[21] Marco Ortu, Giuseppe Destefanis, Bram Adams, Alessandro Murgia, Michele Marchesi, and Roberto Tonelli. 2015. The JIRA Repository Dataset: Understand-