How Does Contributors' Involvement Influence the Build Status of an Open-Source Software Project?

Marcel Rebouças

mscr@cin.ufpe.br

Renato Olivera

ros3@cin.ufpe.br

Gustavo Pinto

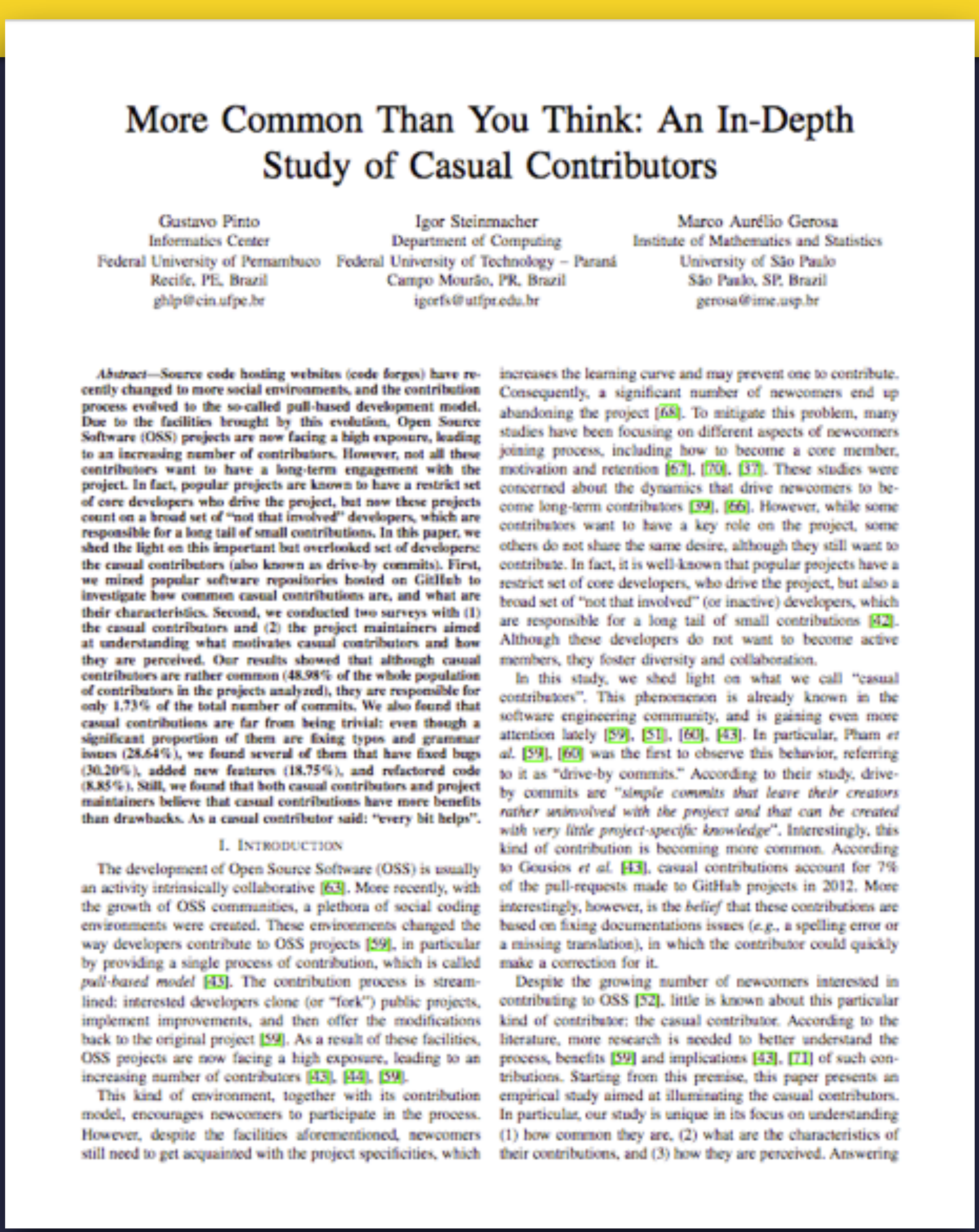
gpinto@ufpa.br

Fernando Castor

castor@cin.ufpe.br

Motivation (1/3)

Casual contributors represent a **large** part of the OSS community (49%) and their contributions are **far from trivial.**



More Common Than You Think: An In-Depth Study of Casual Contributors

Gustavo Pinto
Informatics Center
Federal University of Pernambuco
Recife, PE, Brazil
ghlp@cin.ufpe.br

Igor Steinmacher
Department of Computing
Federal University of Technology – Paraná
Campo Mourão, PR, Brazil
igorfs@utfpr.edu.br

Marco Aurélio Gerosa
Institute of Mathematics and Statistics
University of São Paulo
São Paulo, SP, Brazil
gerosa@ime.usp.br

***Abstract*—Source code hosting websites (code forges) have recently changed to more social environments, and the contribution process evolved to the so-called pull-based development model. Due to the facilities brought by this evolution, Open Source Software (OSS) projects are now facing a high exposure, leading to an increasing number of contributors. However, not all these contributors want to have a long-term engagement with the project. In fact, popular projects are known to have a restrict set of core developers who drive the project, but now these projects count on a broad set of "not that involved" developers, which are responsible for a long tail of small contributions. In this paper, we shed the light on this important but overlooked set of developers: the casual contributors (also known as drive-by commits). First, we mined popular software repositories hosted on GitHub to investigate how common casual contributions are, and what are their characteristics. Second, we conducted two surveys with (1) the casual contributors and (2) the project maintainers aimed at understanding what motivates casual contributors and how they are perceived. Our results showed that although casual contributors are rather common (48.98% of the whole population of contributors in the projects analyzed), they are responsible for only 1.73% of the total number of commits. We also found that casual contributions are far from being trivial: even though a significant proportion of them are fixing types and grammar issues (28.64%), we found several of them that have fixed bugs (30.20%), added new features (18.75%), and refactored code (8.85%). Still, we found that both casual contributors and project maintainers believe that casual contributions have more benefits than drawbacks. As a casual contributor said: "every bit helps".**

I. Introduction

The development of Open Source Software (OSS) is usually an activity intrinsically collaborative [63]. More recently, with the growth of OSS communities, a plethora of social coding environments were created. These environments changed the way developers contribute to OSS projects [59], in particular by providing a single process of contribution, which is called *pull-based model* [43]. The contribution process is streamlined: interested developers clone (or "fork") public projects, implement improvements, and then offer the modifications back to the original project [59]. As a result of these facilities, OSS projects are now facing a high exposure, leading to an increasing number of contributors [43], [44], [59].

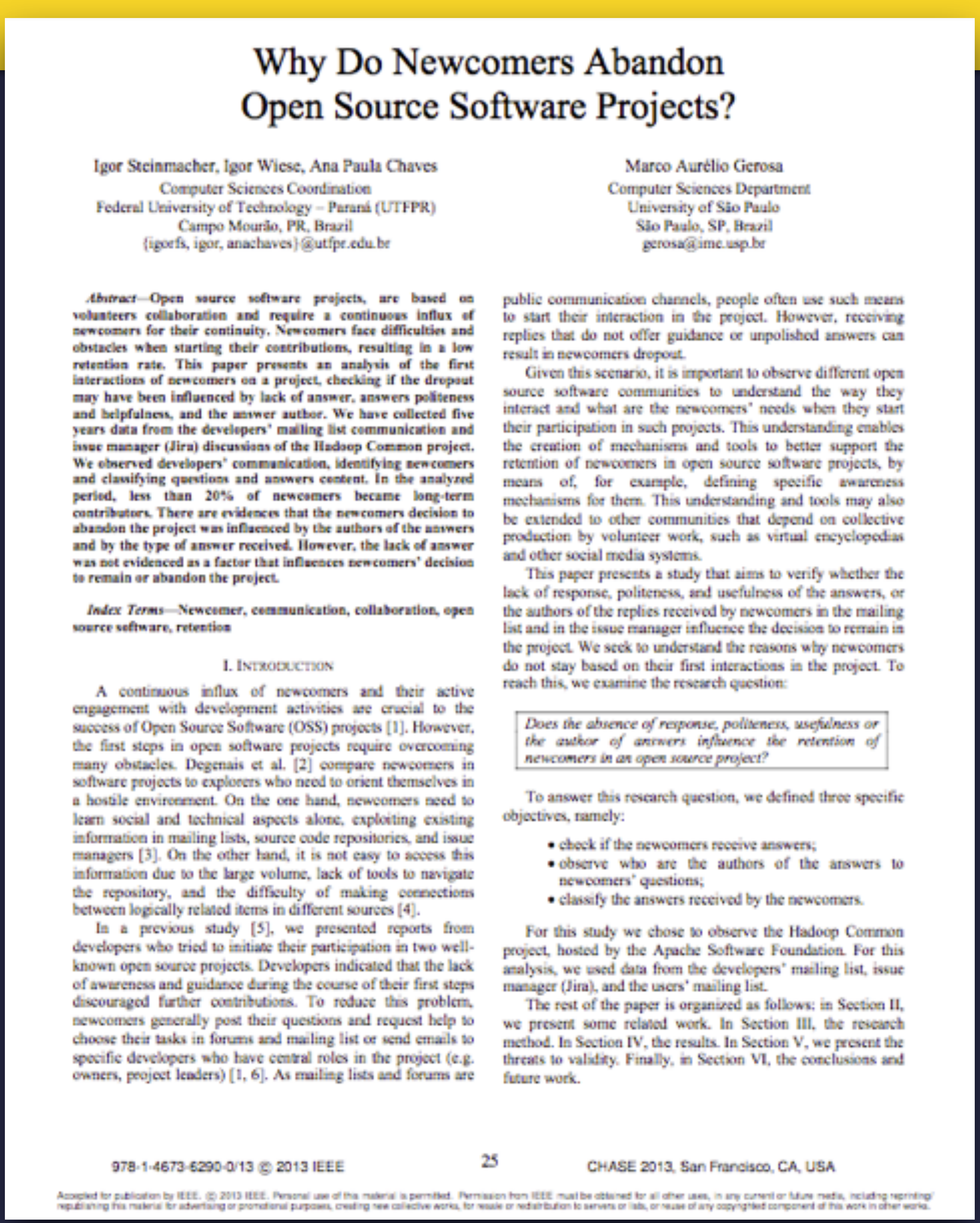
This kind of environment, together with its contribution model, encourages newcomers to participate in the process. However, despite the facilities aforementioned, newcomers still need to get acquainted with the project specificities, which increases the learning curve and may prevent one to contribute. Consequently, a significant number of newcomers end up abandoning the project [68]. To mitigate this problem, many studies have been focusing on different aspects of newcomers joining process, including how to become a core member, motivation and retention [67], [70], [37]. These studies were concerned about the dynamics that drive newcomers to become long-term contributors [39], [66]. However, while some contributors want to have a key role on the project, some others do not share the same desire, although they still want to contribute. In fact, it is well-known that popular projects have a restrict set of core developers, who drive the project, but also a broad set of "not that involved" (or inactive) developers, which are responsible for a long tail of small contributions [42]. Although these developers do not want to become active members, they foster diversity and collaboration.

In this study, we shed light on what we call "casual contributors". This phenomenon is already known in the software engineering community, and is gaining even more attention lately [59], [51], [60], [43]. In particular, Pham *et al.* [59], [60] was the first to observe this behavior, referring to it as "drive-by commits." According to their study, drive-by commits are "*simple commits that leave their creators rather uninvolved with the project and that can be created with very little project-specific knowledge*". Interestingly, this kind of contribution is becoming more common. According to Gousios *et al.* [43], casual contributions account for 7% of the pull-requests made to GitHub projects in 2012. More interestingly, however, is the *belief* that these contributions are based on fixing documentations issues (*e.g.*, a spelling error or a missing translation), in which the contributor could quickly make a correction for it.

Despite the growing number of newcomers interested in contributing to OSS [52], little is known about this particular kind of contributor: the casual contributor. According to the literature, more research is needed to better understand the process, benefits [59] and implications [43], [71] of such contributions. Starting from this premise, this paper presents an empirical study aimed at illuminating the casual contributors. In particular, our study is unique in its focus on understanding (1) how common they are, (2) what are the characteristics of their contributions, and (3) how they are perceived. Answering

Motivation (2/3)

Only 20% of the new contributors on OSS projects become **long-term** members.



Why Do Newcomers Abandon Open Source Software Projects?

Igor Steinmacher, Igor Wiese, Ana Paula Chaves
Computer Sciences Coordination
Federal University of Technology – Paraná (UTFPR)
Campo Mourão, PR, Brazil
{igorfs, igor, anachaves}@utfpr.edu.br

Marco Aurélio Gerosa
Computer Sciences Department
University of São Paulo
São Paulo, SP, Brazil
gerosa@ime.usp.br

***Abstract*—Open source software projects, are based on volunteers collaboration and require a continuous influx of newcomers for their continuity. Newcomers face difficulties and obstacles when starting their contributions, resulting in a low retention rate. This paper presents an analysis of the first interactions of newcomers on a project, checking if the dropout may have been influenced by lack of answer, answers politeness and helpfulness, and the answer author. We have collected five years data from the developers' mailing list communication and issue manager (Jira) discussions of the Hadoop Common project. We observed developers' communication, identifying newcomers and classifying questions and answers content. In the analyzed period, less than 20% of newcomers became long-term contributors. There are evidences that the newcomers decision to abandon the project was influenced by the authors of the answers and by the type of answer received. However, the lack of answer was not evidenced as a factor that influences newcomers' decision to remain or abandon the project.**

***Index Terms*—Newcomer, communication, collaboration, open source software, retention**

I. Introduction

A continuous influx of newcomers and their active engagement with development activities are crucial to the success of Open Source Software (OSS) projects [1]. However, the first steps in open software projects require overcoming many obstacles. Degenais et al. [2] compare newcomers in software projects to explorers who need to orient themselves in a hostile environment. On the one hand, newcomers need to learn social and technical aspects alone, exploiting existing information in mailing lists, source code repositories, and issue managers [3]. On the other hand, it is not easy to access this information due to the large volume, lack of tools to navigate the repository, and the difficulty of making connections between logically related items in different sources [4].

In a previous study [5], we presented reports from developers who tried to initiate their participation in two well-known open source projects. Developers indicated that the lack of awareness and guidance during the course of their first steps discouraged further contributions. To reduce this problem, newcomers generally post their questions and request help to choose their tasks in forums and mailing list or send emails to specific developers who have central roles in the project (e.g. owners, project leaders) [1, 6]. As mailing lists and forums are public communication channels, people often use such means to start their interaction in the project. However, receiving replies that do not offer guidance or unpolished answers can result in newcomers dropout.

Given this scenario, it is important to observe different open source software communities to understand the way they interact and what are the newcomers' needs when they start their participation in such projects. This understanding enables the creation of mechanisms and tools to better support the retention of newcomers in open source software projects, by means of, for example, defining specific awareness mechanisms for them. This understanding and tools may also be extended to other communities that depend on collective production by volunteer work, such as virtual encyclopedias and other social media systems.

This paper presents a study that aims to verify whether the lack of response, politeness, and usefulness of the answers, or the authors of the replies received by newcomers in the mailing list and in the issue manager influence the decision to remain in the project. We seek to understand the reasons why newcomers do not stay based on their first interactions in the project. To reach this, we examine the research question:

Does the absence of response, politeness, usefulness or the author of answers influence the retention of newcomers in an open source project?

To answer this research question, we defined three specific objectives, namely:

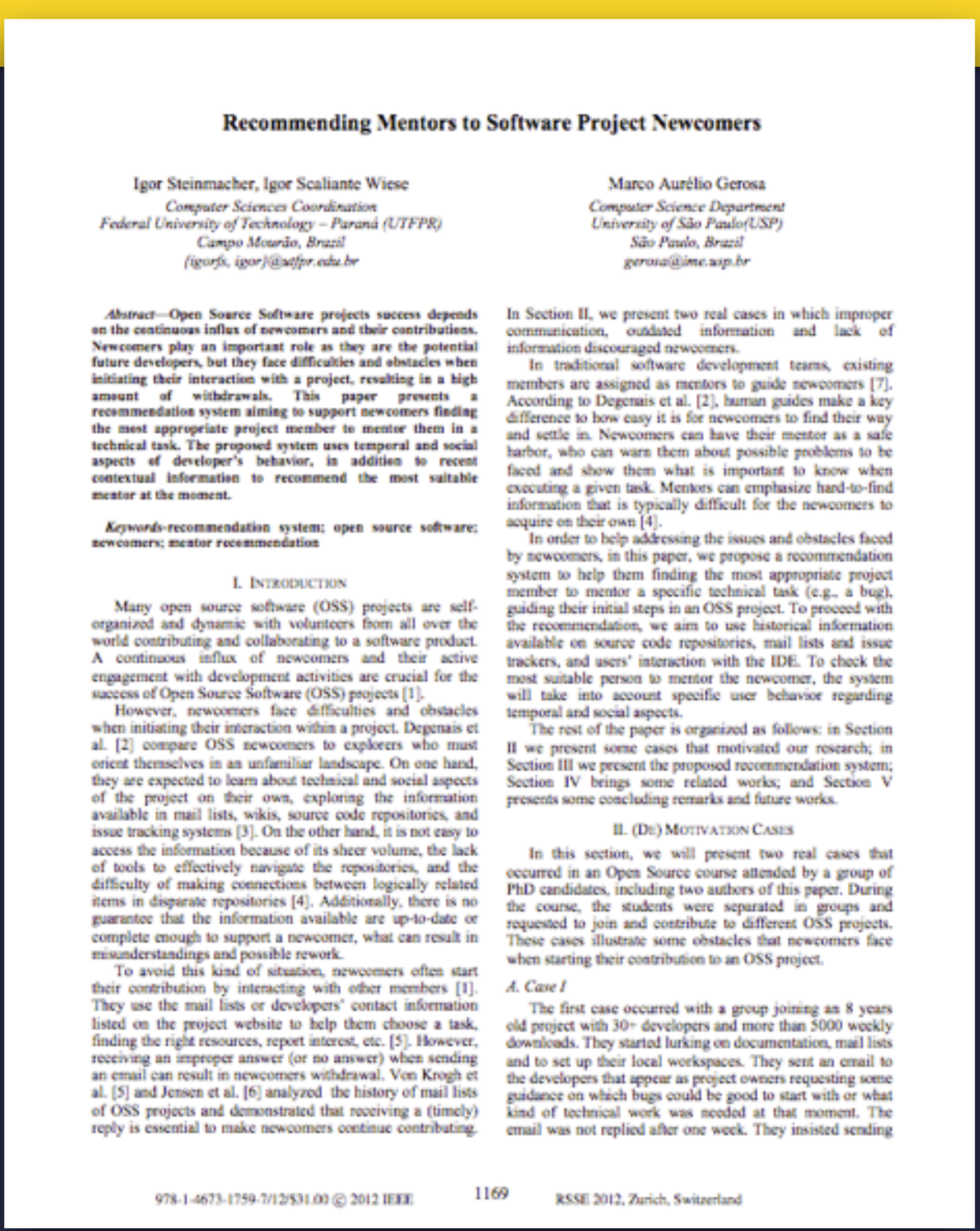
- check if the newcomers receive answers;
- observe who are the authors of the answers to newcomers' questions;
- classify the answers received by the newcomers.

For this study we chose to observe the Hadoop Common project, hosted by the Apache Software Foundation. For this analysis, we used data from the developers' mailing list, issue manager (Jira), and the users' mailing list.

The rest of the paper is organized as follows: in Section II, we present some related work. In Section III, the research method. In Section IV, the results. In Section V, we present the threats to validity. Finally, in Section VI, the conclusions and future work.

Motivation (3/3)

Lack of awareness and guidance during their first steps makes it **hard** to contribute!



Recommending Mentors to Software Project Newcomers

Igor Steinmacher, Igor Scaliante Wiese
Computer Sciences Coordination
Federal University of Technology – Paraná (UTFPR)
Campo Mourão, Brazil
{igorfs, igor}@utfpr.edu.br

Marco Aurélio Gerosa
Computer Science Department
University of São Paulo(USP)
São Paulo, Brazil
gerosa@ime.usp.br

***Abstract*—Open Source Software projects success depends on the continuous influx of newcomers and their contributions. Newcomers play an important role as they are the potential future developers, but they face difficulties and obstacles when initiating their interaction with a project, resulting in a high amount of withdrawals. This paper presents a recommendation system aiming to support newcomers finding the most appropriate project member to mentor them in a technical task. The proposed system uses temporal and social aspects of developer's behavior, in addition to recent contextual information to recommend the most suitable mentor at the moment.**

***Keywords*-recommendation system; open source software; newcomers; mentor recommendation**

I. Introduction

Many open source software (OSS) projects are self-organized and dynamic with volunteers from all over the world contributing and collaborating to a software product. A continuous influx of newcomers and their active engagement with development activities are crucial for the success of Open Source Software (OSS) projects [1].

However, newcomers face difficulties and obstacles when initiating their interaction within a project. Degenais et al. [2] compare OSS newcomers to explorers who must orient themselves in an unfamiliar landscape. On one hand, they are expected to learn about technical and social aspects of the project on their own, exploring the information available in mail lists, wikis, source code repositories, and issue tracking systems [3]. On the other hand, it is not easy to access the information because of its sheer volume, the lack of tools to effectively navigate the repositories, and the difficulty of making connections between logically related items in disparate repositories [4]. Additionally, there is no guarantee that the information available are up-to-date or complete enough to support a newcomer, what can result in misunderstandings and possible rework.

To avoid this kind of situation, newcomers often start their contribution by interacting with other members [1]. They use the mail lists or developers' contact information listed on the project website to help them choose a task, finding the right resources, report interest, etc. [5]. However, receiving an improper answer (or no answer) when sending an email can result in newcomers withdrawal. Von Krogh et al. [5] and Jensen et al. [6] analyzed the history of mail lists of OSS projects and demonstrated that receiving a (timely) reply is essential to make newcomers continue contributing. In Section II, we present two real cases in which improper communication, outdated information and lack of information discouraged newcomers.

In traditional software development teams, existing members are assigned as mentors to guide newcomers [7]. According to Degenais et al. [2], human guides make a key difference to how easy it is for newcomers to find their way and settle in. Newcomers can have their mentor as a safe harbor, who can warn them about possible problems to be faced and show them what is important to know when executing a given task. Mentors can emphasize hard-to-find information that is typically difficult for the newcomers to acquire on their own [4].

In order to help addressing the issues and obstacles faced by newcomers, in this paper, we propose a recommendation system to help them finding the most appropriate project member to mentor a specific technical task (e.g., a bug), guiding their initial steps in an OSS project. To proceed with the recommendation, we aim to use historical information available on source code repositories, mail lists and issue trackers, and users' interaction with the IDE. To check the most suitable person to mentor the newcomer, the system will take into account specific user behavior regarding temporal and social aspects.

The rest of the paper is organized as follows: in Section II we present some cases that motivated our research; in Section III we present the proposed recommendation system; Section IV brings some related works; and Section V presents some concluding remarks and future works.

II. (De) Motivation Cases

In this section, we will present two real cases that occurred in an Open Source course attended by a group of PhD candidates, including two authors of this paper. During the course, the students were separated in groups and requested to join and contribute to different OSS projects. These cases illustrate some obstacles that newcomers face when starting their contribution to an OSS project.

A. Case I

The first case occurred with a group joining an 8 years old project with 30+ developers and more than 5000 weekly downloads. They started lurking on documentation, mail lists and to set up their local workspaces. They sent an email to the developers that appear as project owners requesting some guidance on which bugs could be good to start with or what kind of technical work was needed at that moment. The email was not replied after one week. They insisted sending

978-1-4673-1759-7/12/$31.00 © 2012 IEEE

RSSE 2012, Zurich, Switzerland

Casual Contributors

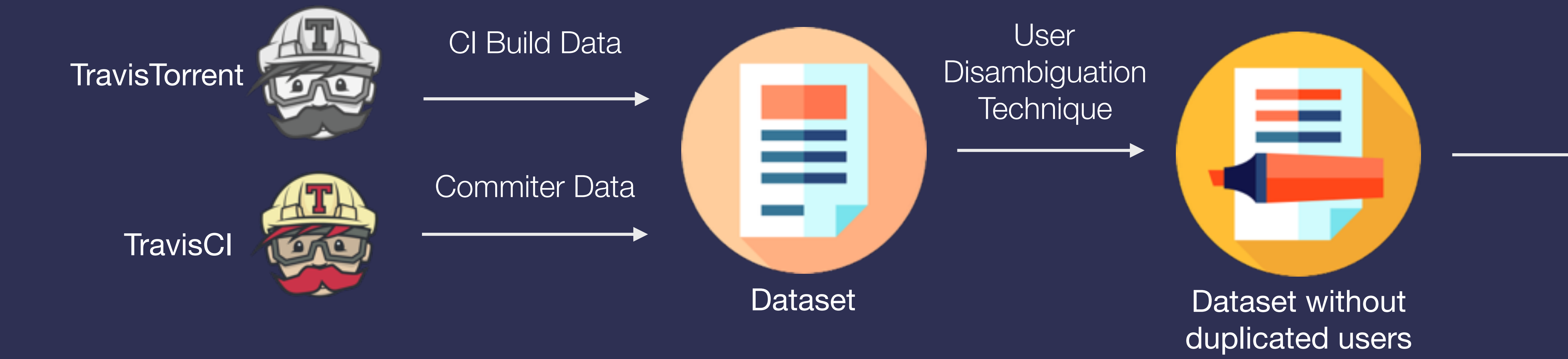
They have little or no prior knowledge on the project domain

They might put more effort to create their first pull-request

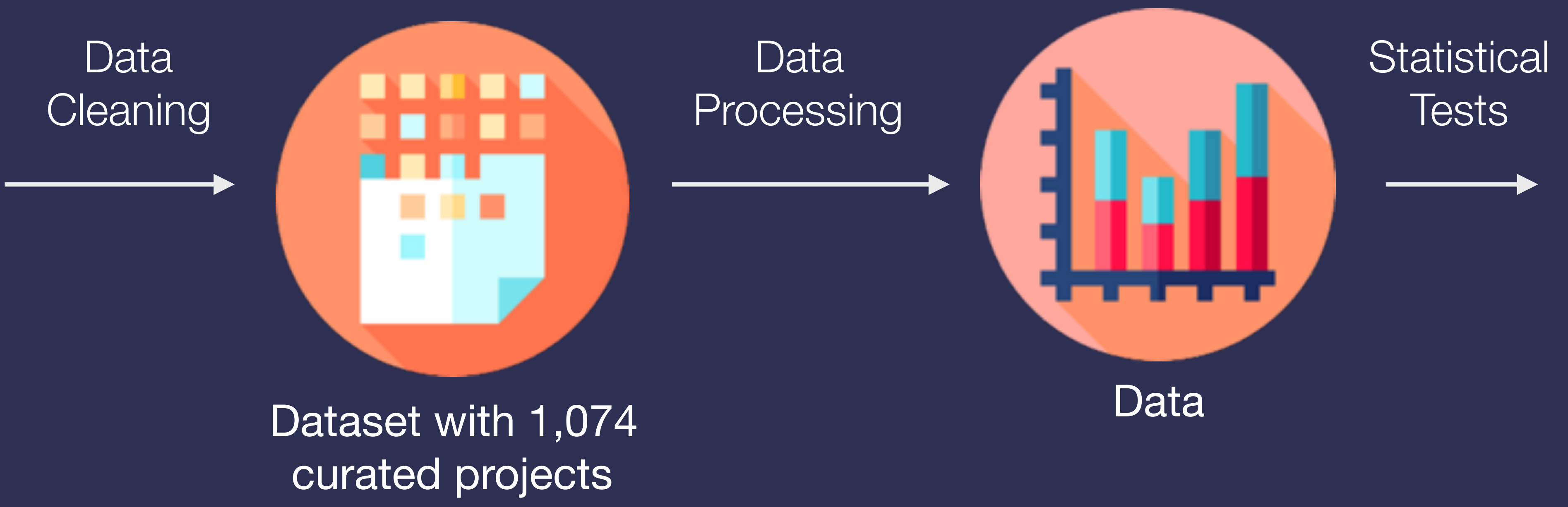
RESEARCH QUESTION

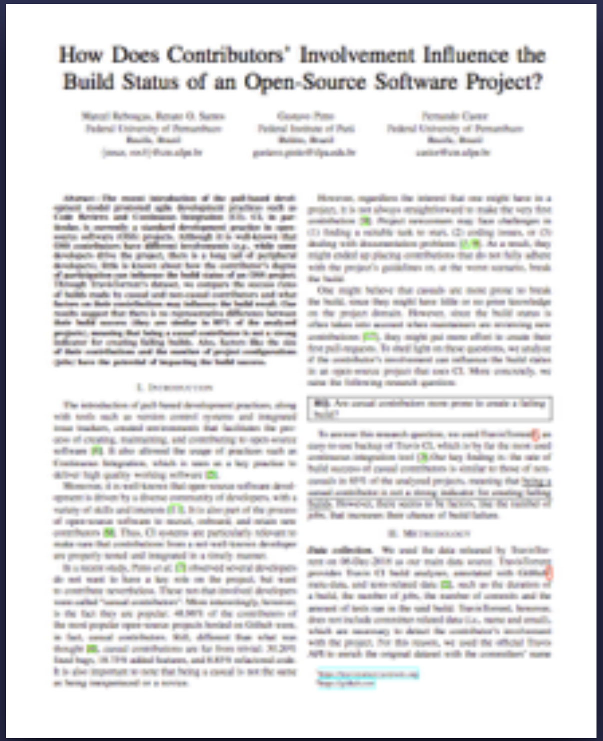
Are casual contributors more prone to create a failing build?

Methodology



TravisTorrent
TravisCI
CI Build Data
Commiter Data
Dataset
User Disambiguation Technique
Dataset without duplicated users



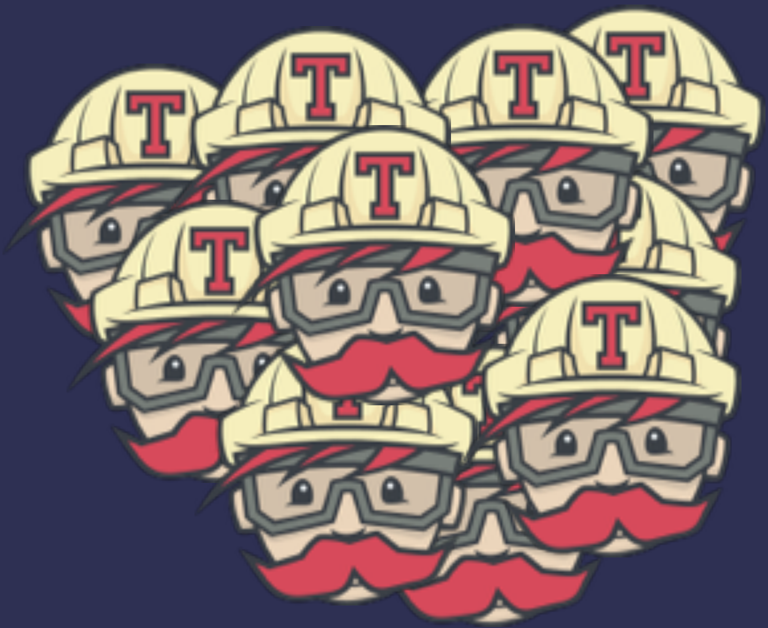


How Does Contributors' Involvement Influence the Build Status of an Open-Source Software Project?

Data Overview

1,074 projects

35,360 users

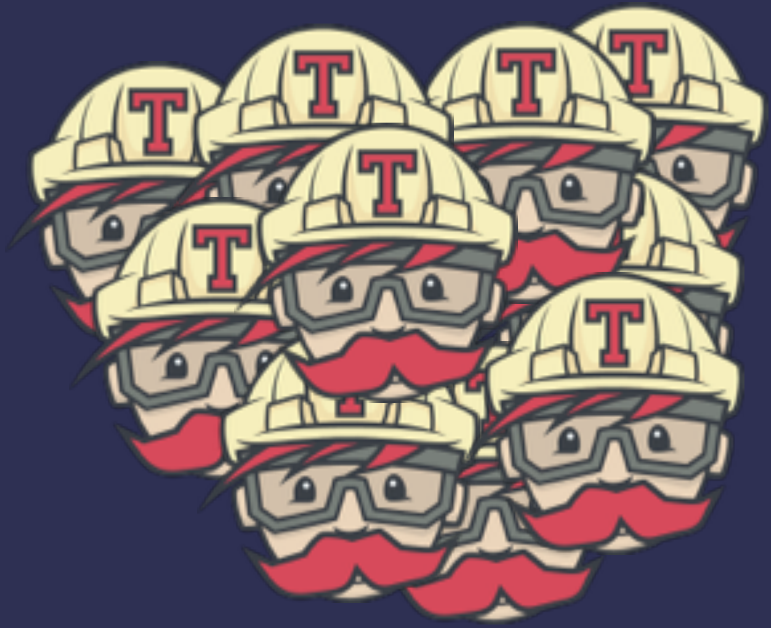


619,370 builds

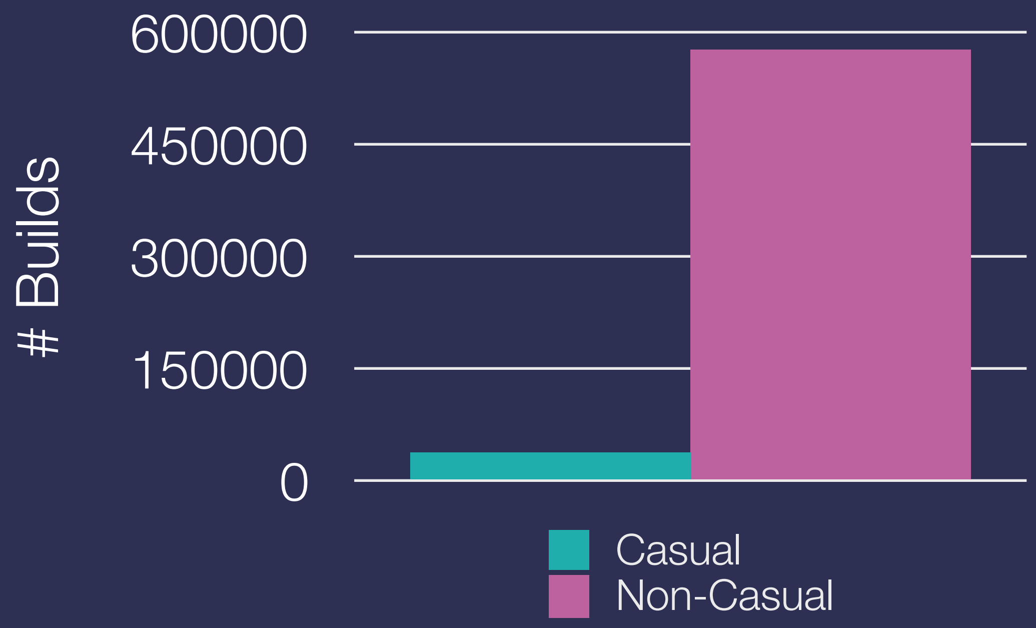
Data Overview

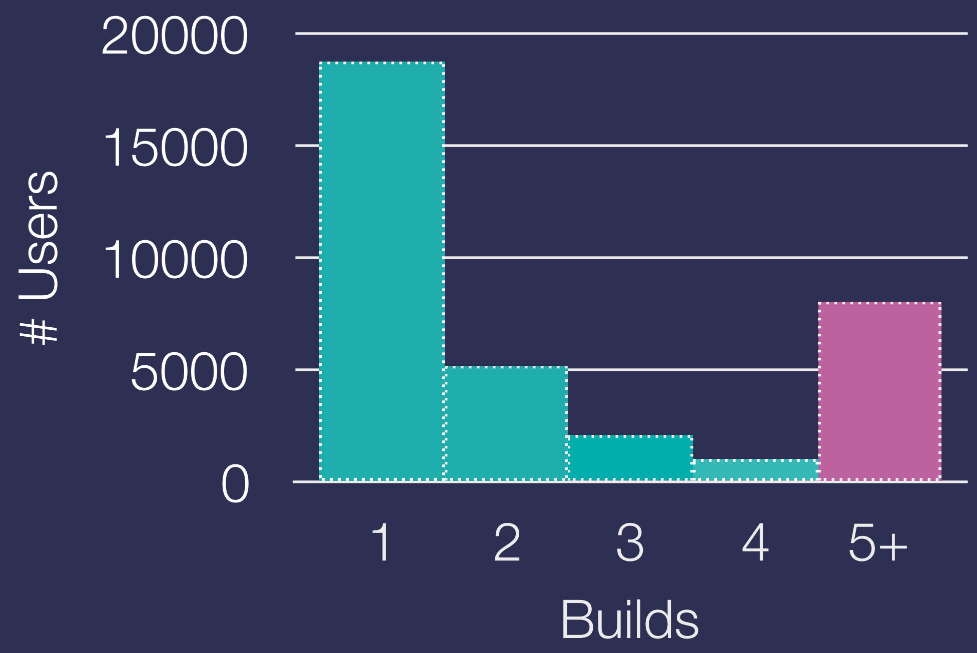
1,074 projects

35,360 users



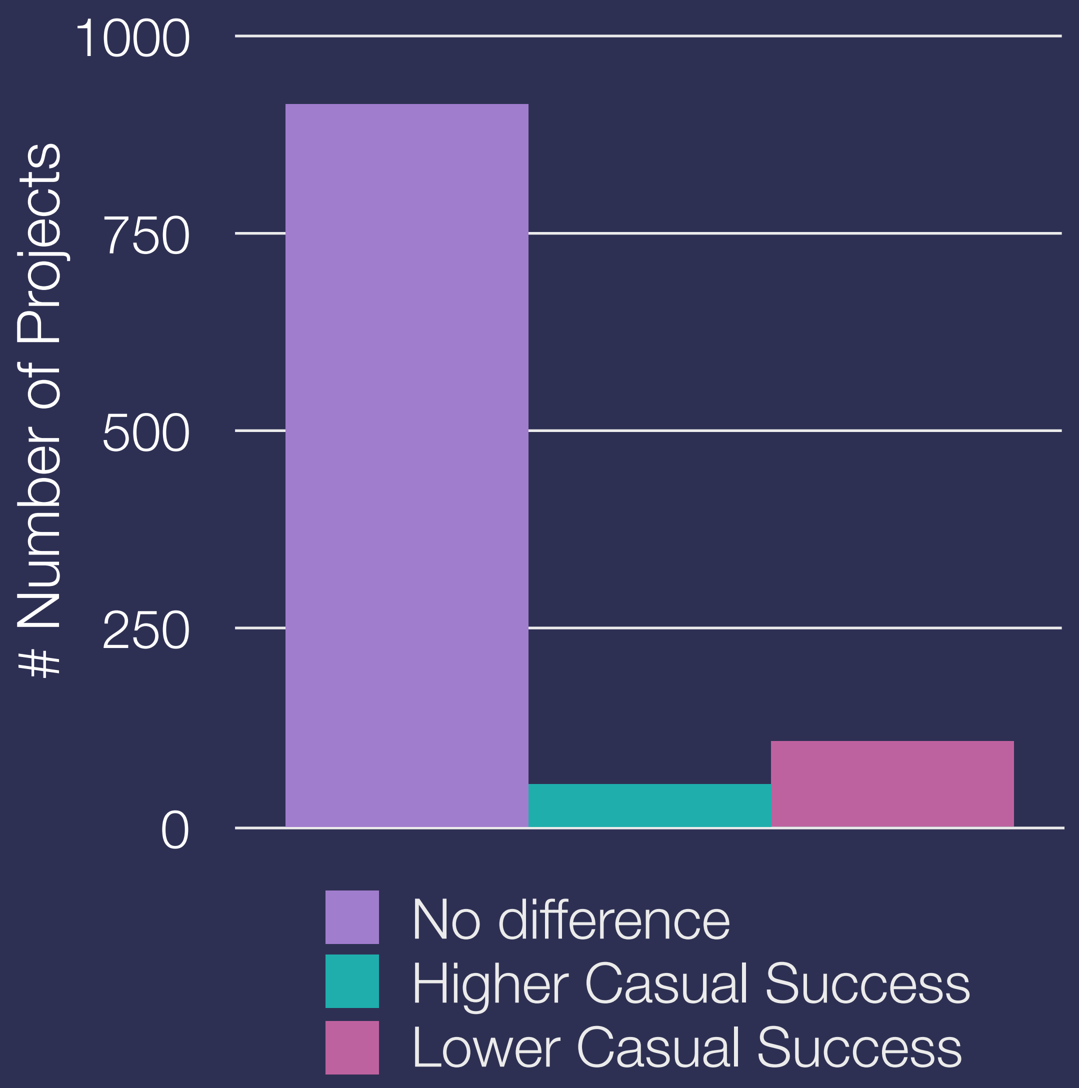
619,370 builds





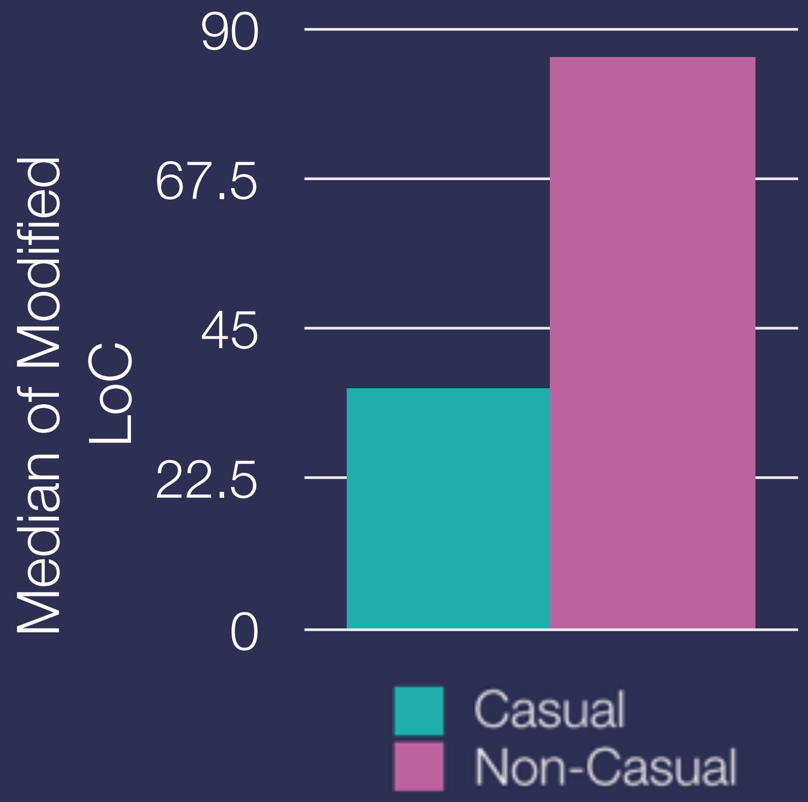
Results

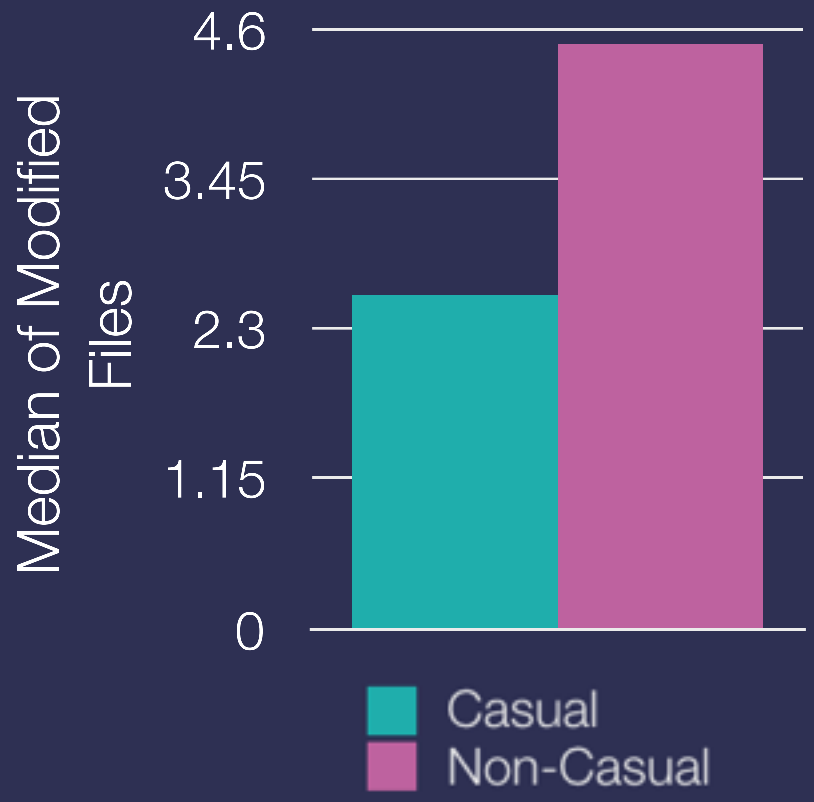
Being a casual contributor is not a strong indicator for creating failing builds



Results

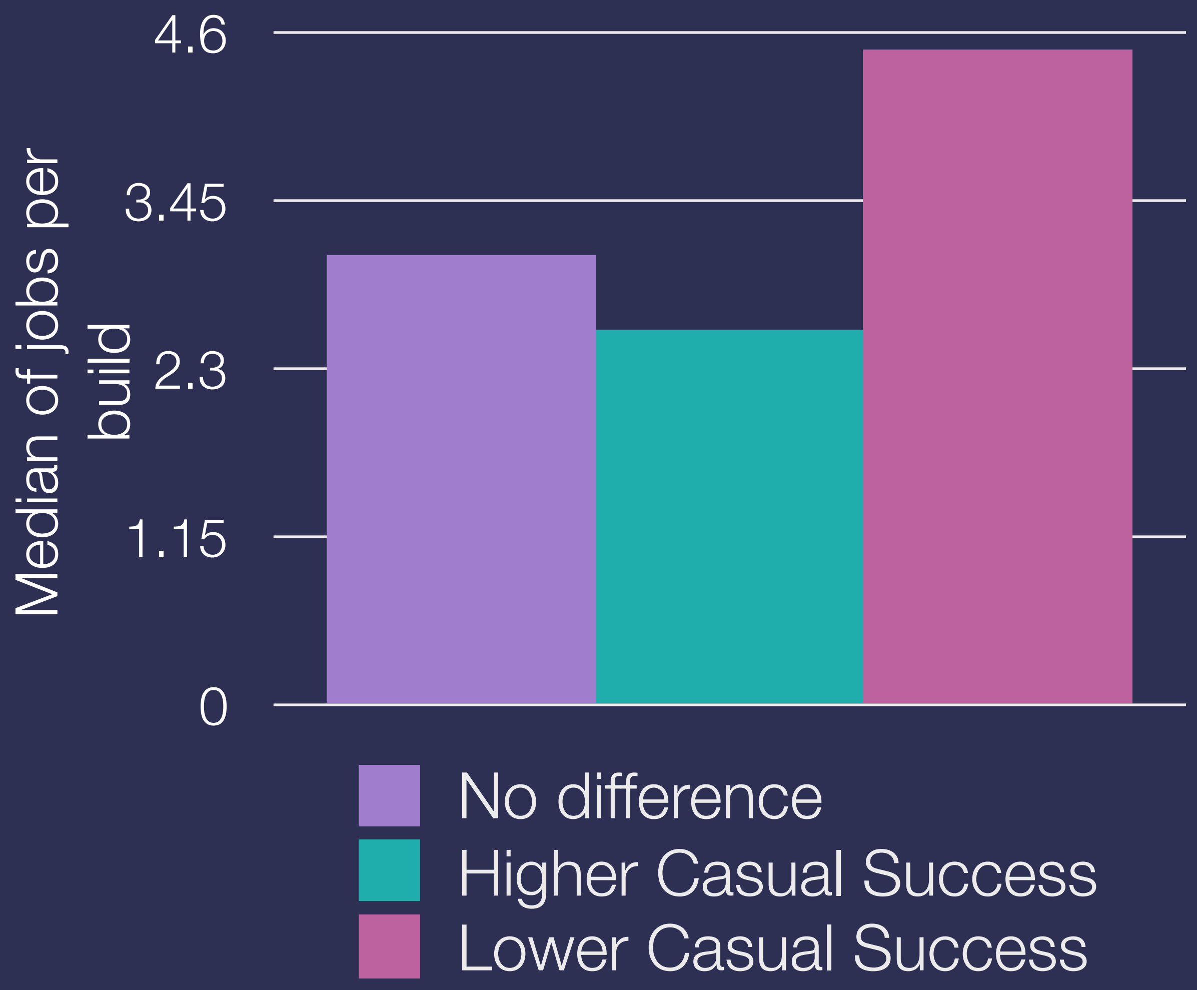
Casual contributions are smaller, both in modified source-code lines and modified files





Results

Projects in which casuals fail more than non-casuals run **more jobs** per build.



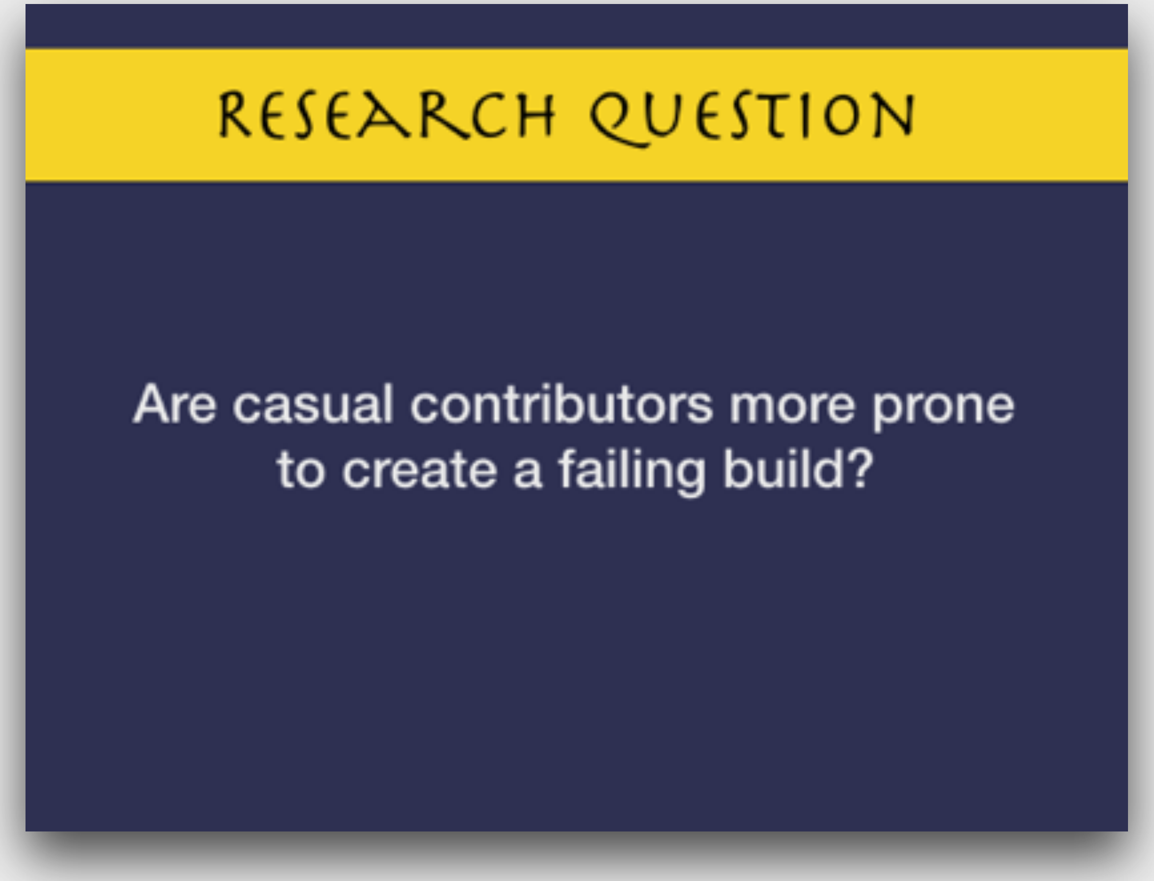
Take-Away Message

Are you a casual contributor?

Take-Away Message

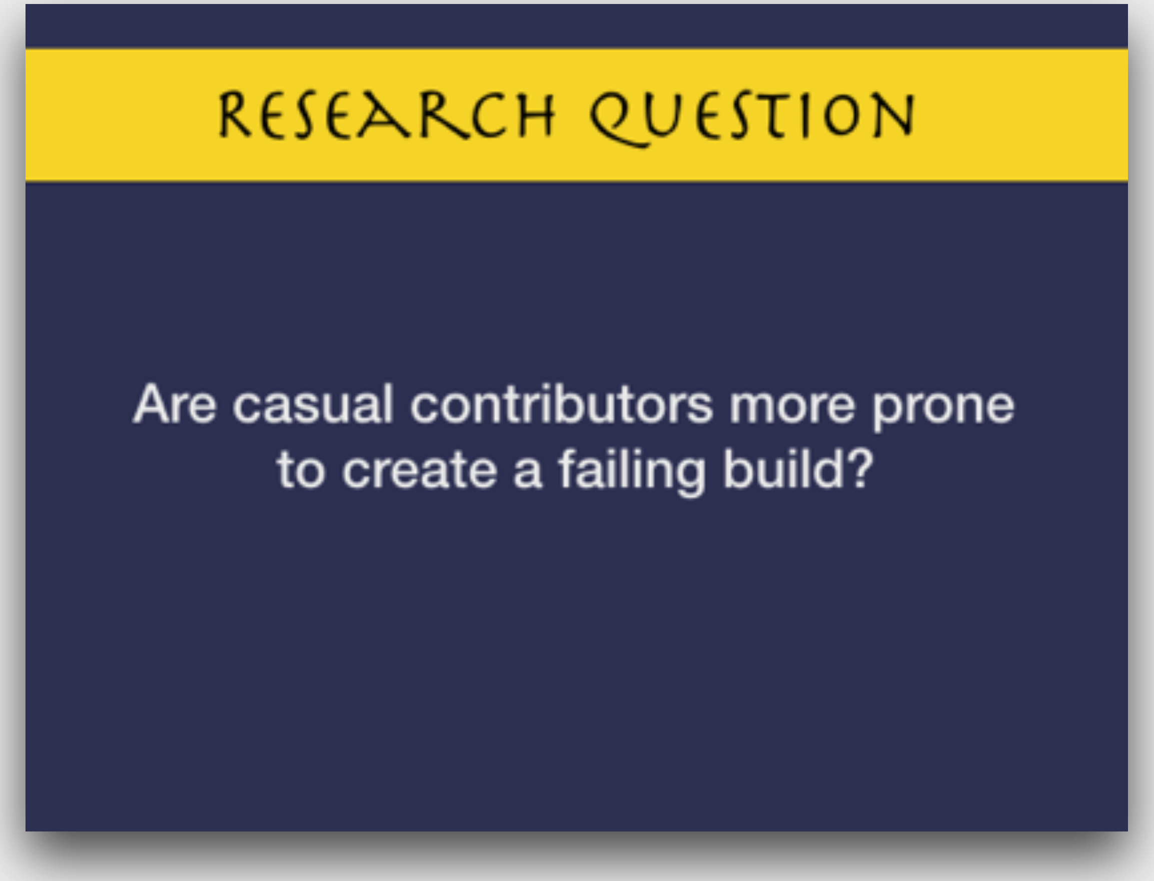
Are you a casual contributor?

Go ahead and contribute!



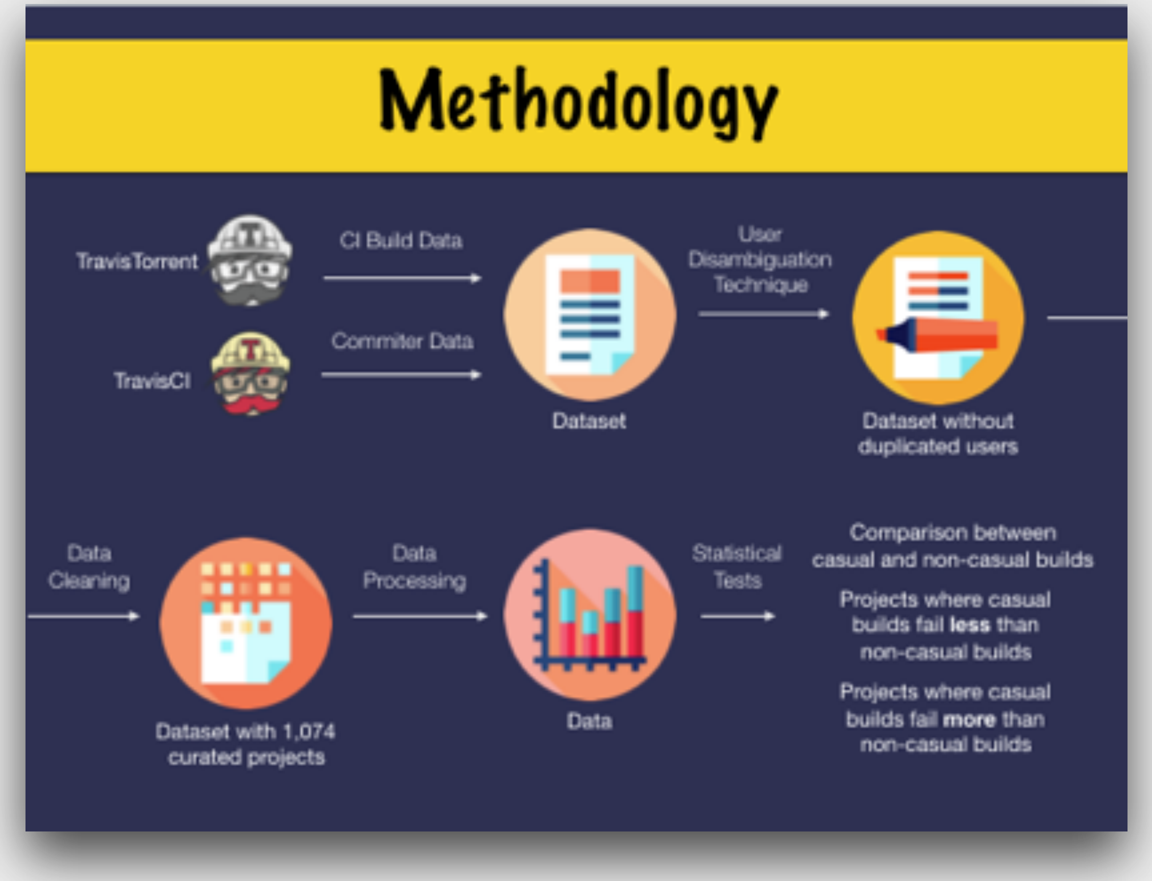
RESEARCH QUESTION

Are casual contributors more prone to create a failing build?



Research Question

Are casual contributors more prone to create a failing build?

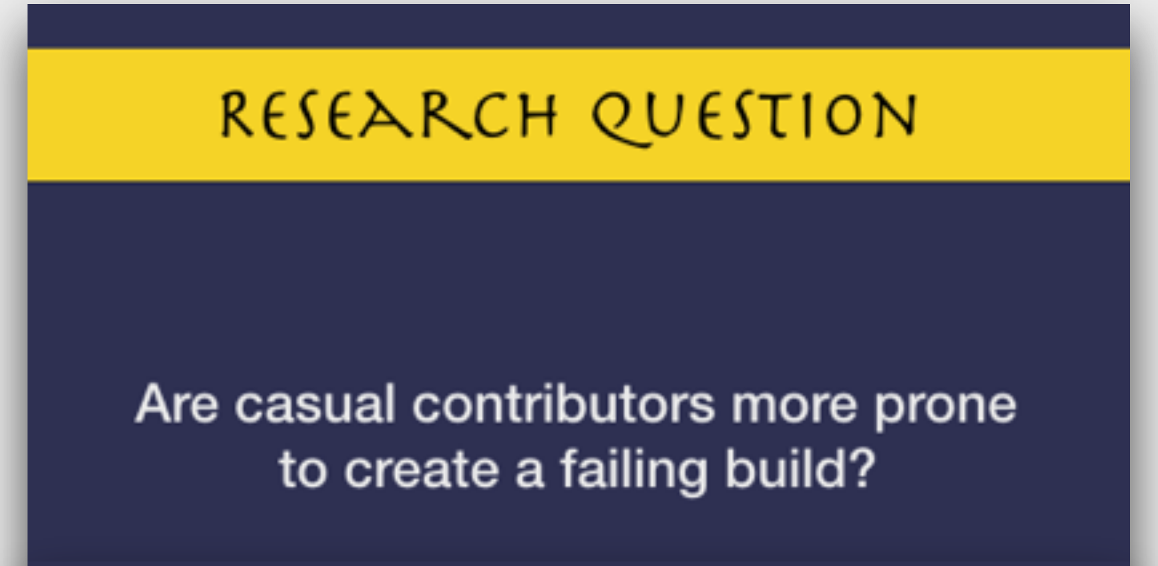


Methodology

Comparison between casual and non-casual builds

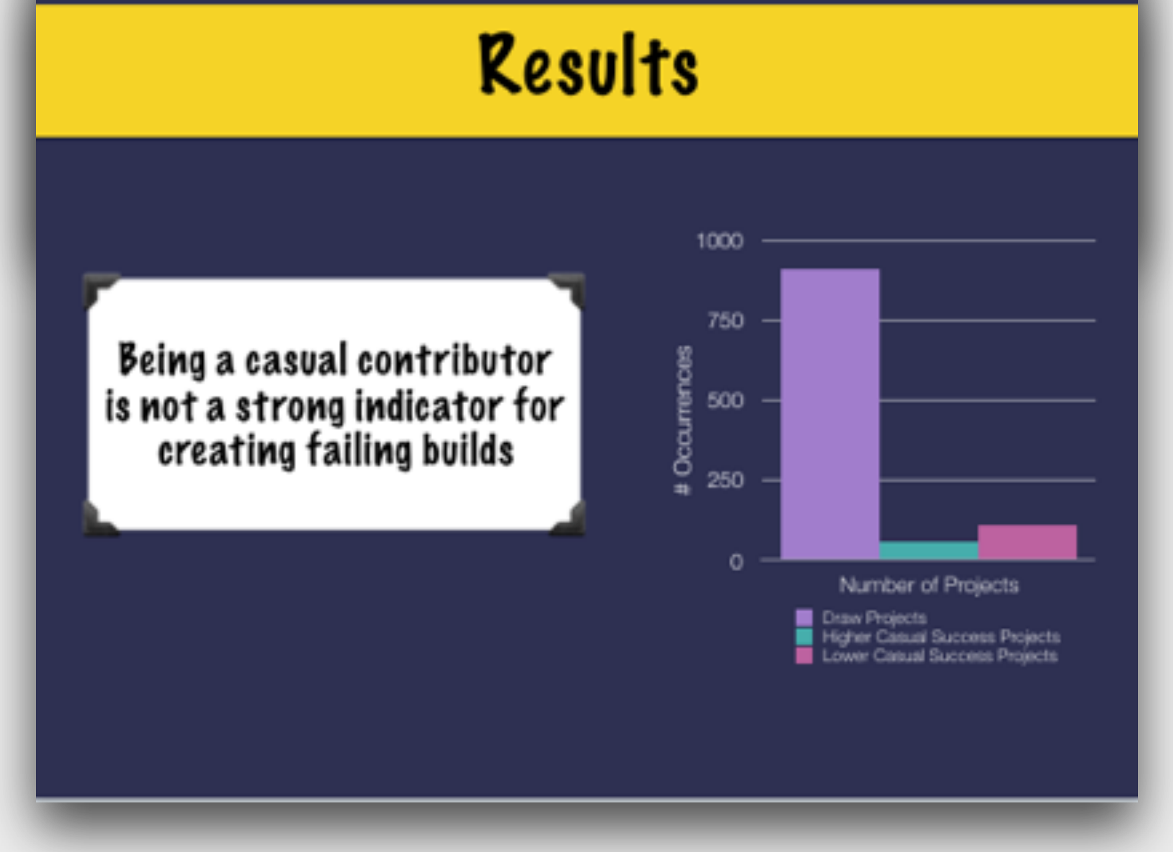
Projects where casual builds fail **less** than non-casual builds

Projects where casual builds fail **more** than non-casual builds



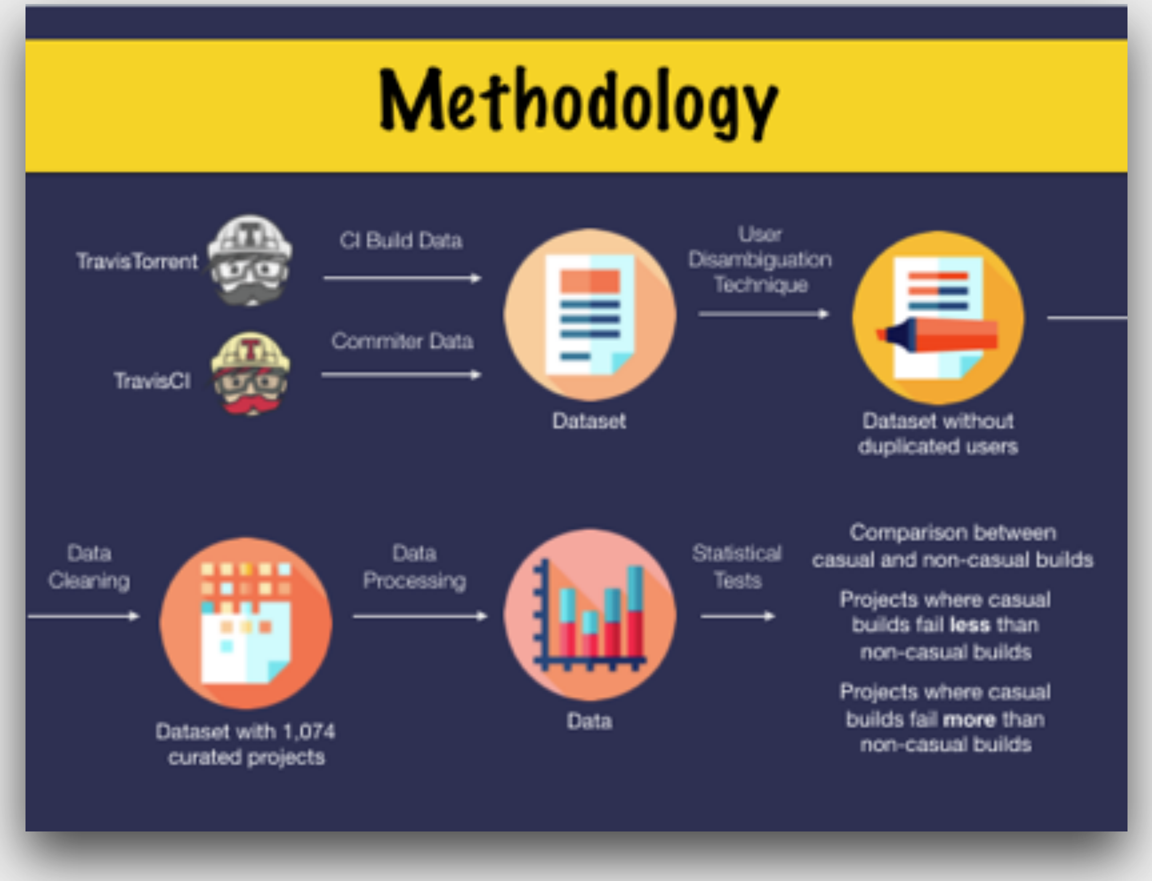
RESEARCH QUESTION

Are casual contributors more prone to create a failing build?



Results

Being a casual contributor is not a strong indicator for creating failing builds



Methodology

TravisTorrent

TravisCI

CI Build Data

Commiter Data

Dataset

User Disambiguation Technique

Dataset without duplicated users

Data Cleaning

Dataset with 1,074 curated projects

Data Processing

Data

Statistical Tests

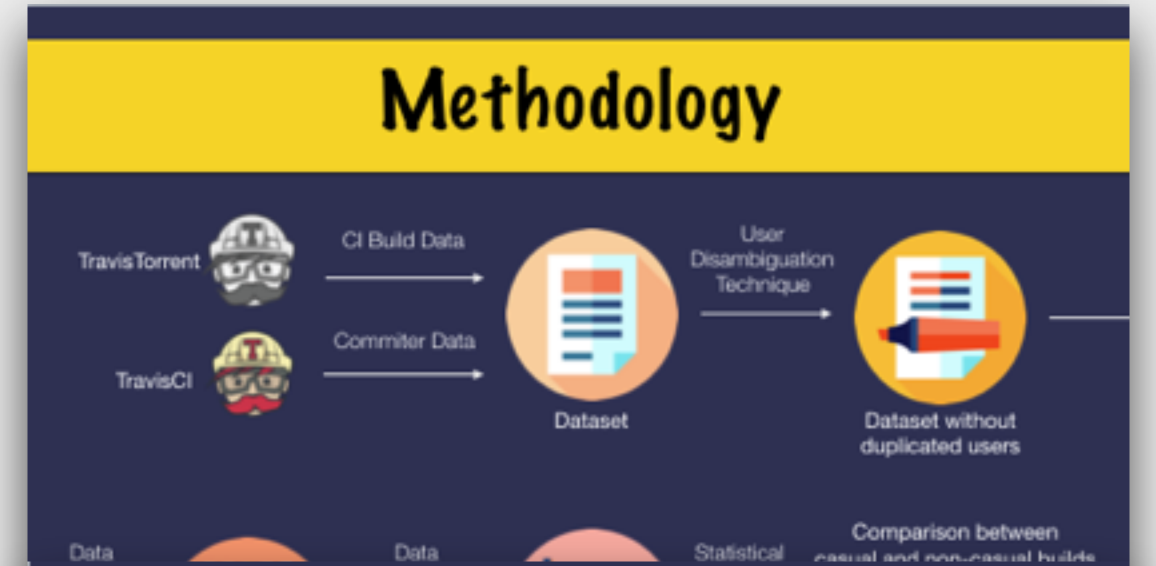
Comparison between casual and non-casual builds

Projects where casual builds fail **less** than non-casual builds

Projects where casual builds fail **more** than non-casual builds

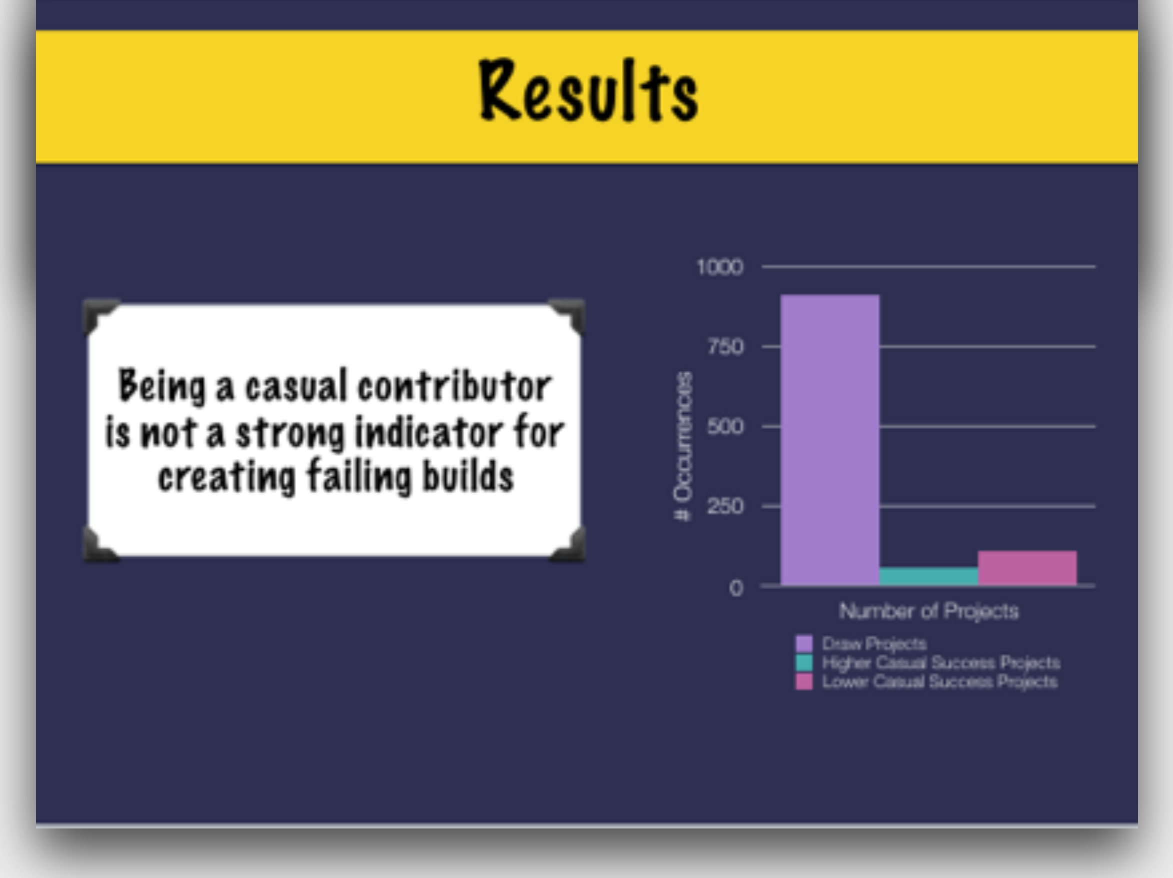
RESEARCH QUESTION

Are casual contributors more prone to create a failing build?



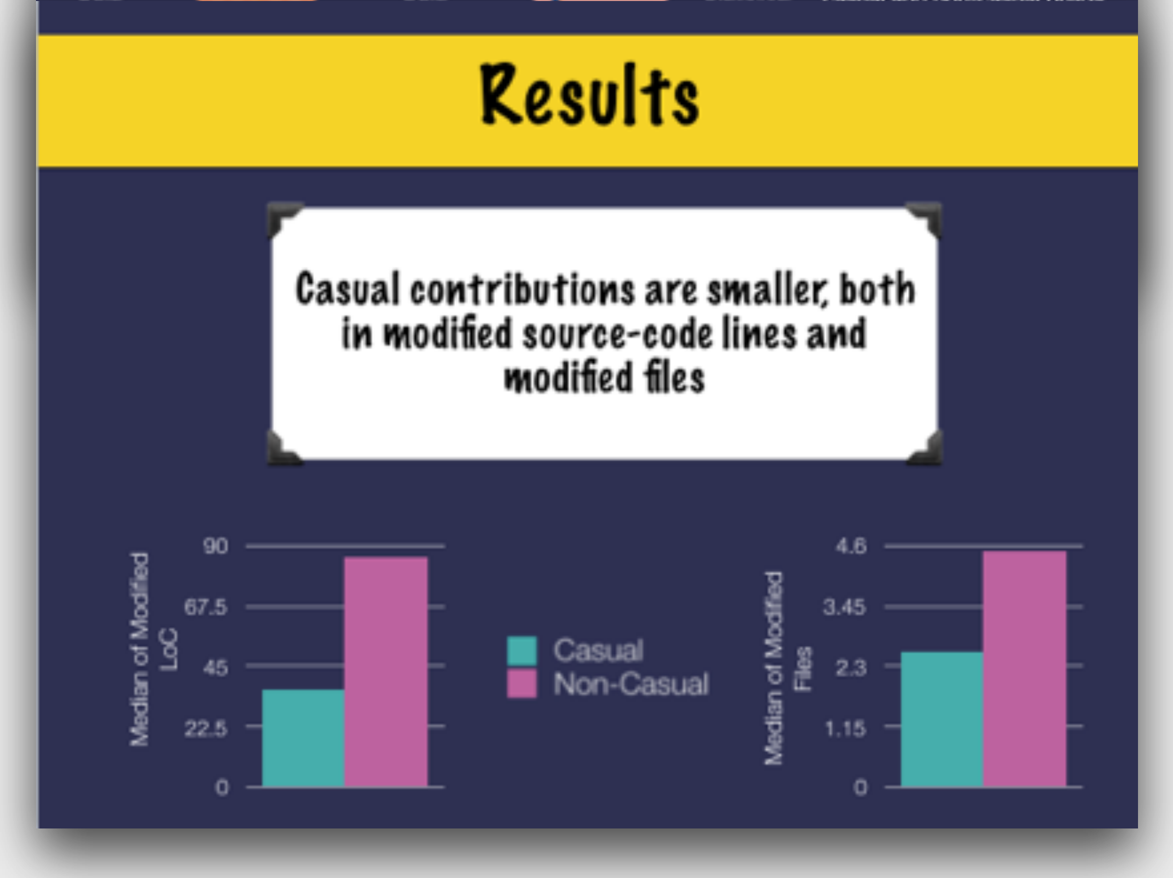
Methodology

TravisTorrent
TravisCI
CI Build Data
Commiter Data
Dataset
User Disambiguation Technique
Dataset without duplicated users
Data
Data
Statistical
Comparison between casual and non-casual builds



Results

Being a casual contributor is not a strong indicator for creating failing builds



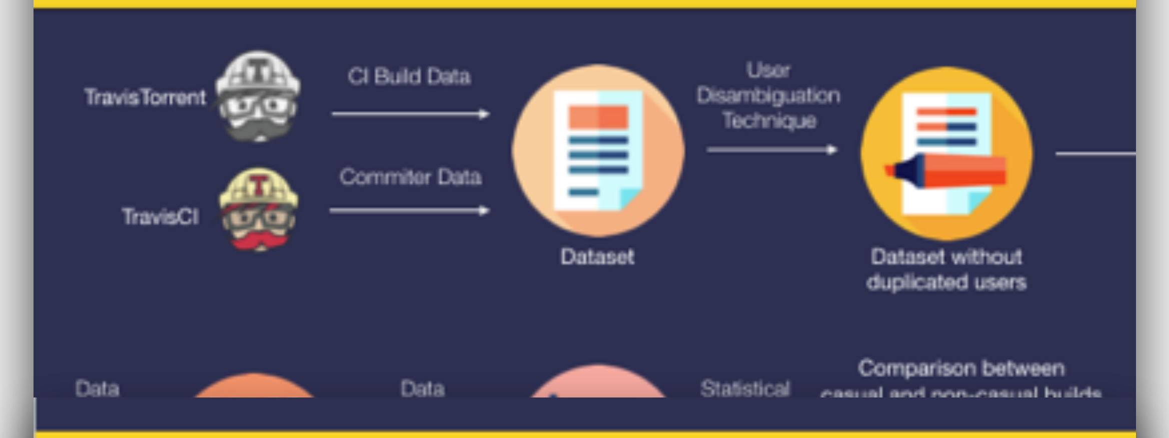
Results

Casual contributions are smaller, both in modified source-code lines and modified files

Research Question

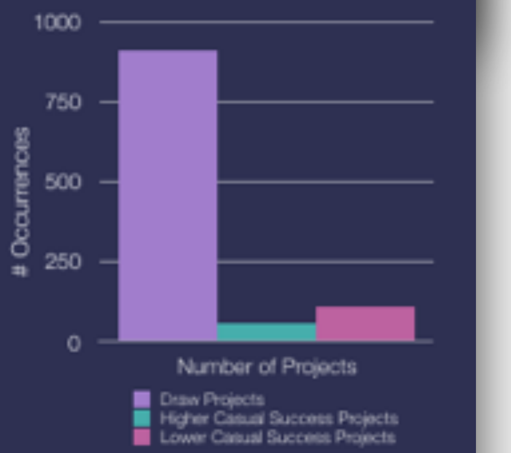
Are casual contributors more prone to create a failing build?

Methodology


TravisTorrent
TravisCI
CI Build Data
Commiter Data
Dataset
User Disambiguation Technique
Dataset without duplicated users
Data
Data
Statistical
Comparison between casual and non-casual builds

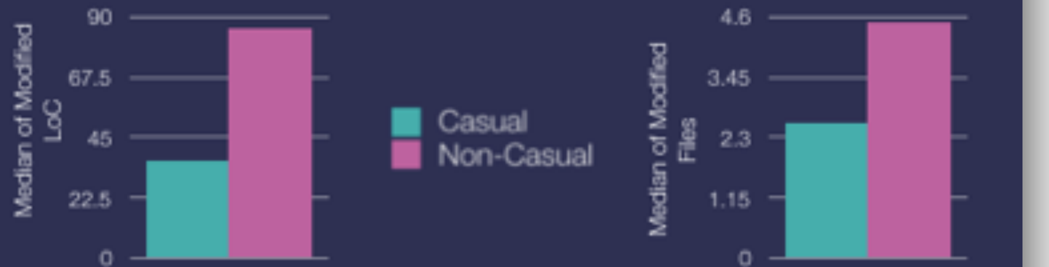
Results

Being a casual contributor is not a strong indicator for creating failing builds



Results

Casual contributions are smaller, both in modified source-code lines and modified files



ANY QUESTIONS?

@gustavopinto

gpinto@ufpa.br