MSR '19, May 26–27, Montréal, QC, Canada

Does UML™ Modeling Associate with Lower Defect Proneness?

⚠WARNING SPOILER ALERT

Slightly

Adithya Raghuraman
STRUDEL @CMU

Truong Ho-Quang
Chalmers U of Tech

Michel Chaudron
Chalmers U of Tech

Alexander Serebrenik
TU Eindhoven

Bogdan Vasilescu
STRUDEL @CMU

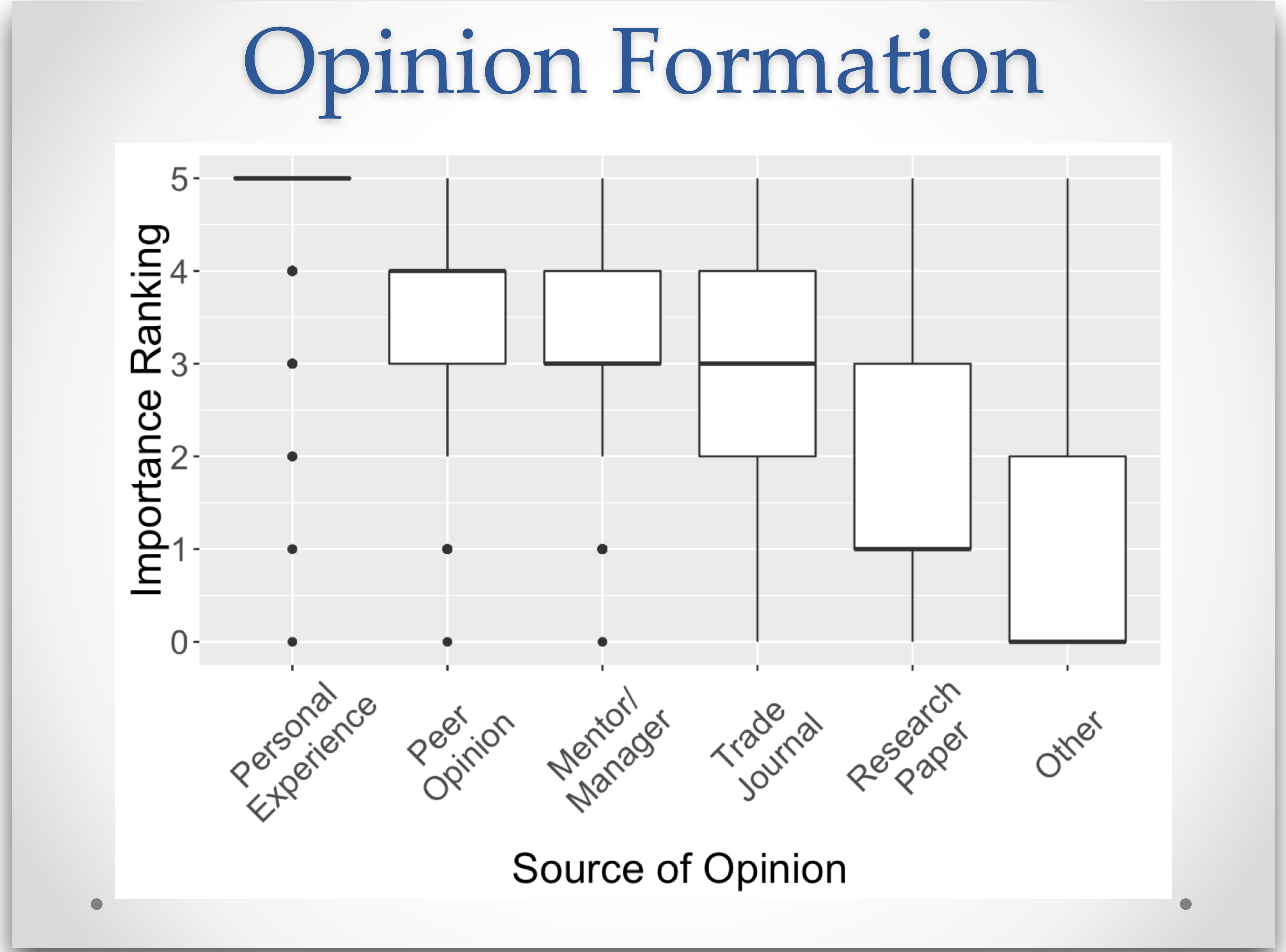
STRUDEL
SOCIO-TECHNICAL RESEARCH USING DATA EXCAVATION LAB
CHALMERS
Carnegie Mellon University
TU/e EINDHOVEN UNIVERSITY OF TECHNOLOGY
isr institute for SOFTWARE RESEARCH

Belief vs evidence

40 percent of major decisions are based not on facts, but on the **manager's gut.**

Accenture survey among 254 US managers in industry.
http://newsroom.accenture.com/article_display.cfm?article_id=4777

Microsoft Research



(Devanbu et al, ICSE 2016)

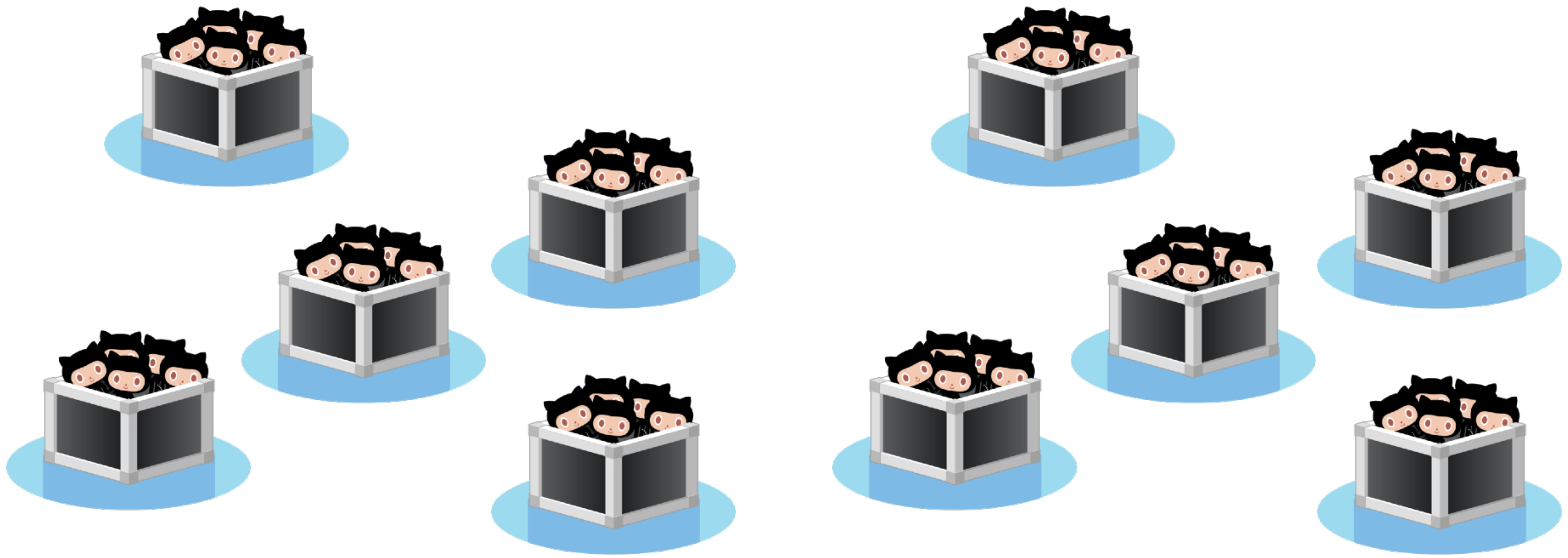
We revisit a widely-held belief:

Does the use of UML modeling, on average, correlate with higher software quality?

Software design → Team / process maturity and deliberateness → Higher code quality

Natural experiment

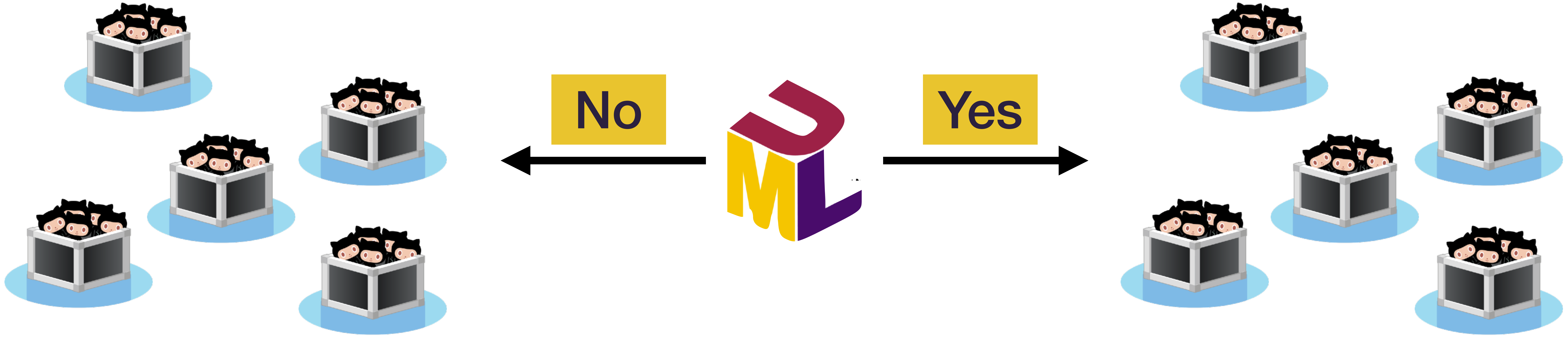
Mine OSS GitHub projects



Natural experiment

"Control" group

"Treatment" group



Natural experiment

(Robles et al, MSR 2017)

An extensive dataset of UML models in GitHub

Gregorio Robles*, Truong Ho-Quang†, Regina Hebig†, Michel R.V. Chaudron†, Miguel Angel Fernandez*
*GSyC/LibreSoft, Universidad Rey Juan Carlos, Madrid, Spain
grex@gsyc.urjc.es, mafesan.nsn@gmail.com
†Chalmers — Göteborg University, Göteborg, Sweden
{truongh, hebig, chaudron}@chalmers.se

***Abstract*—The Unified Modeling Language (UML) is widely taught in academia and has good acceptance in industry. However, there is not an ample dataset of UML diagrams publicly available. Our aim is to offer a dataset of UML files, together with meta-data of the software projects where the UML files belong to. Therefore, we have systematically mined over 12 million GitHub projects to find UML files in them. We present a semi-automated approach to collect UML stored in images, .xmi, and .uml files. We offer a dataset with over 93,000 UML diagrams from over 24,000 projects in GitHub.**

***Keywords*-dataset; UML; GitHub; modeling; mining software repositories;**

I. Introduction

The Unified Modeling Language (UML) provides the facility for software engineers to specify, construct, visualize and document the artifacts of a software-intensive system and to facilitate communication of ideas [1]. UML is commonly taught in the computer science curriculum worldwide, and the use of UML is generally accepted in industrial software development.

However, the number of publicly available examples of UML is relatively low. To the knowledge of the authors, the largest UML dataset up-to-date is the one reported in [2], with around 800 UML models obtained by collecting examples from the literature, web searches, and donations. However, that dataset only contains lone-standing diagram. Thus, it cannot be used for studying the software systems and projects associated to these diagrams.

Even though it has been reported the UML is marginally used in Open Source projects [3][1], the large amount of repositories hosted in GitHub offers the possibility to look for a large number of UML models used in software development projects, together with their source code and development meta-data. This is the reason why we have mined GitHub for UML files. The result of this effort is a dataset with over 93,000 files with UML diagrams. These diagrams comprise several types and formats and offer a valuable data source for educational purposes, as they can be used as real-scenario examples in class, and for further research.

The remainder of this paper is structured as follows: Next, we introduce how we have extracted the data. Section III possibilities that such a dataset offers to researchers and practitioners. After presenting future improvements in Section V, we detail the limitations and challenges in Section VI. Finally, conclusions are drawn in Section VII

II. Extraction Methodology

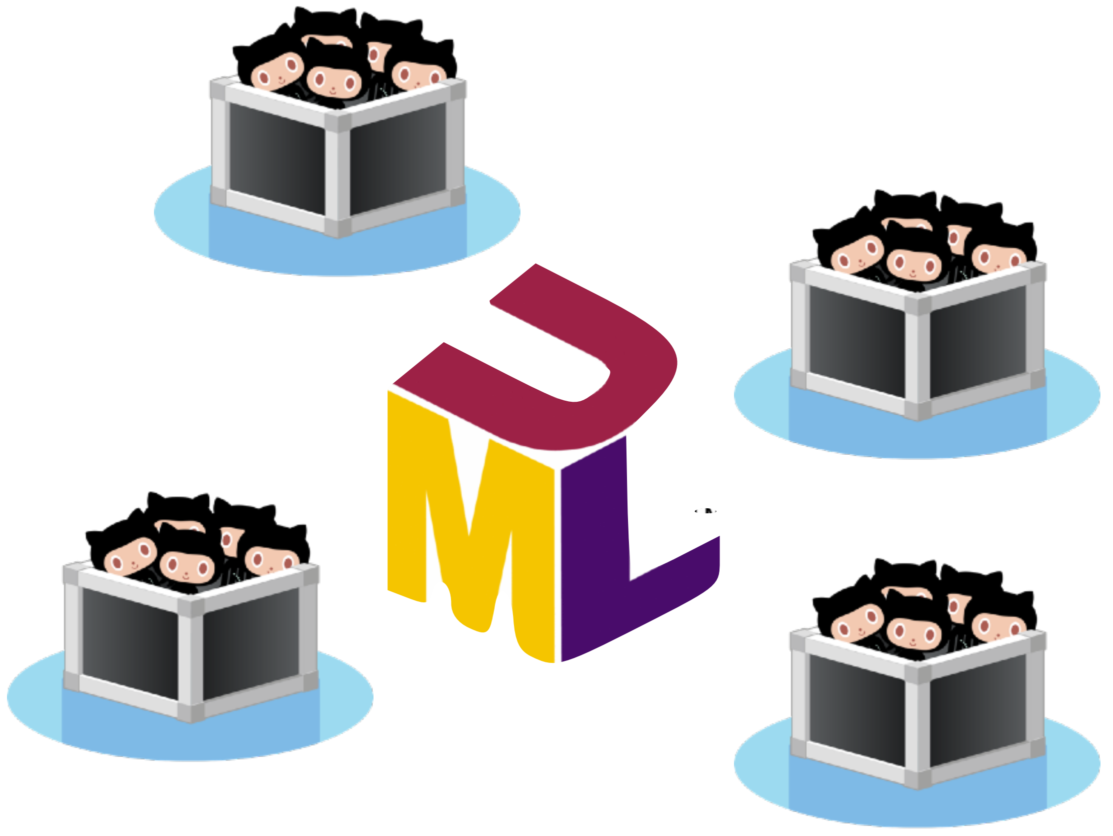
The data extraction process comprises the following four steps: (i) retrieval of the tree (file list) from GitHub repositories (Section II-A), (ii) identification (grepping) of potential UML files (Section II-B), (iii) automated examination (and manual evaluation) of the existence of UML notation in the obtained files (Section II-C), and (iv) retrieval of the meta-data from those repositories where a UML file has been identified (Section II-D).

A. Step 1: Mining GitHub

We depart with a list of GitHub repositories obtained from GHTorrent [4][2], which offers a list of over 15M non-forked non-deleted repositories. Since GHTorrent now distributes CSV files (one file per table) instead of mysqldump based backups, we use data available in the projects.csv file: the URL of the project and the values of *forked_from* and *deleted* (as we discard those projects that are forks or have been removed/deleted).

For those projects that are not forks nor have been deleted, we retrieve from the GitHub API[3] the tree (file list) for the *master* branch. If the master branch does not exist, then we query again the GitHub API for the branch that the project has set as default, and perform a third request to download its tree. With up to three GitHub calls for each repository, given the GitHub API limitation of 5,000 requests/hour, it would take around 14 months to perform the retrieval of data in this first step. As this would have made the data gathering unfeasible, we downloaded the JSON files in parallel with over 20 active GitHub accounts, which were donated during this process. This reduced the time span to approximately one month. For almost 3 million of the repositories we obtained an empty JSON file or an error message from the GitHub API, because the repository has been removed or made private in the time that goes from GHTorrent obtaining its data (which is before February 1st 2016) and our request to the GitHub API (during Summer of 2017).

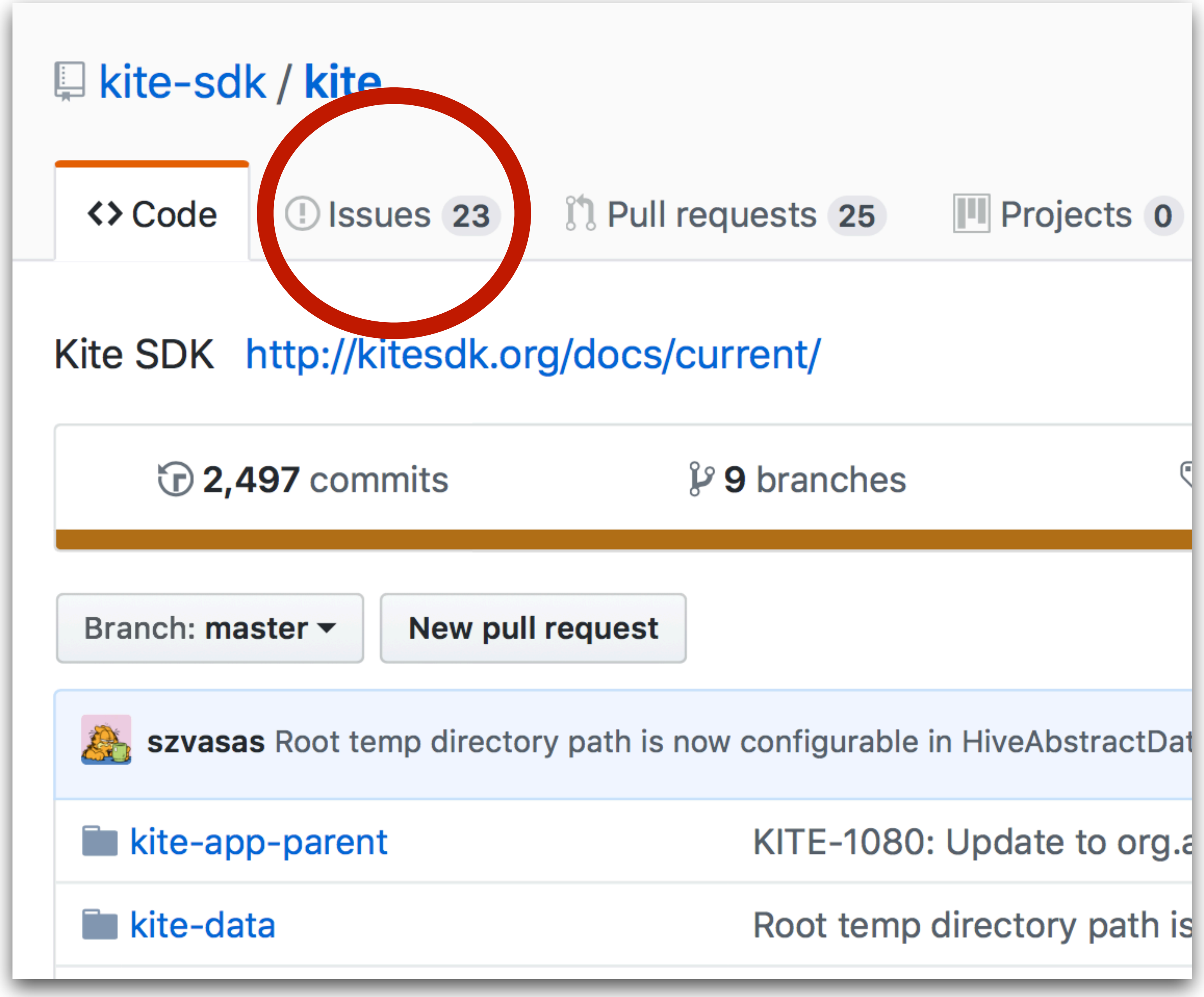
“Treatment” group



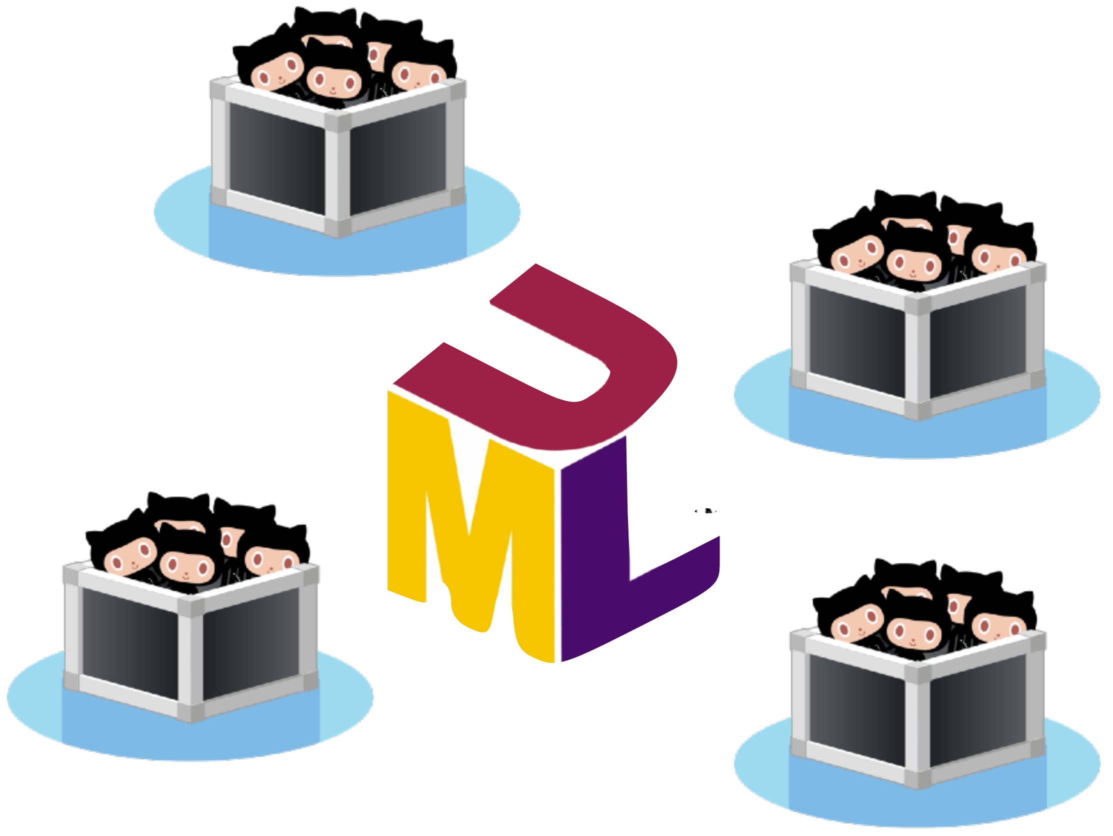
4,650 projects
http://oss.models-db.com

Natural experiment

Mine the
issue tracker



"Treatment" group

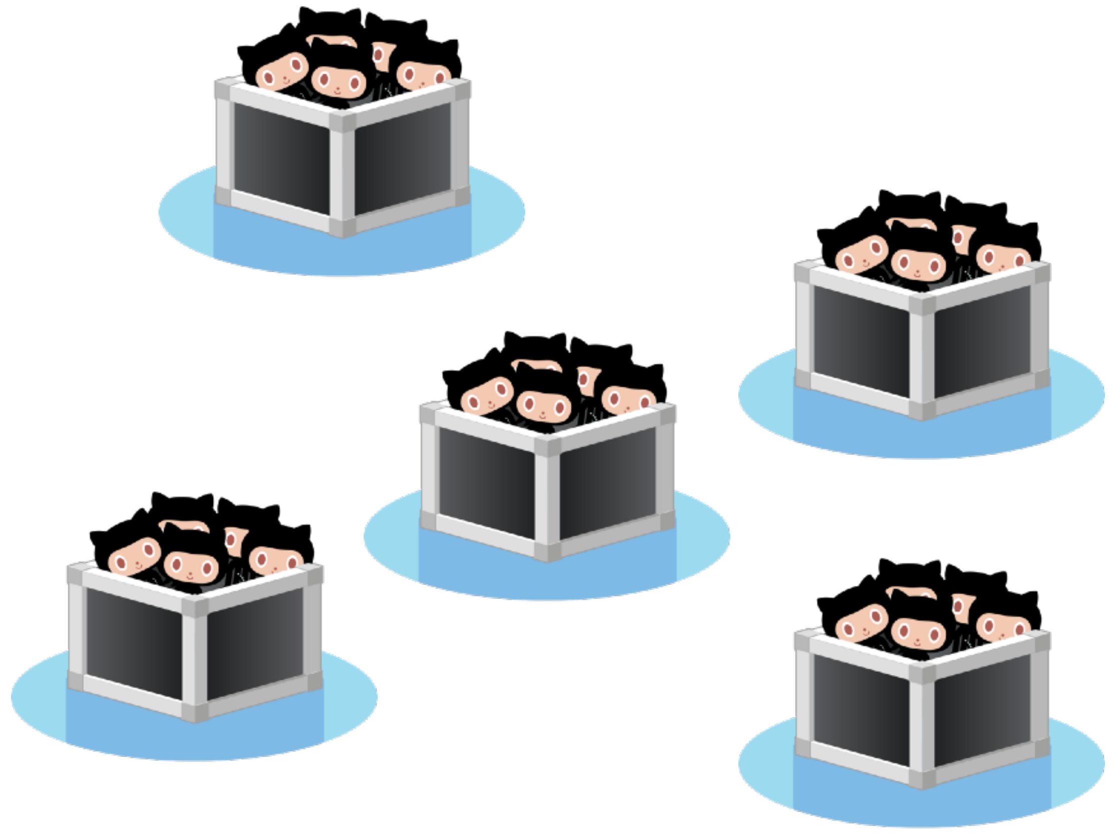


50 projects

- C++, C#, Java
- 2009+
- 10+ stars
- 30+ issues
- Issues in English

Natural experiment

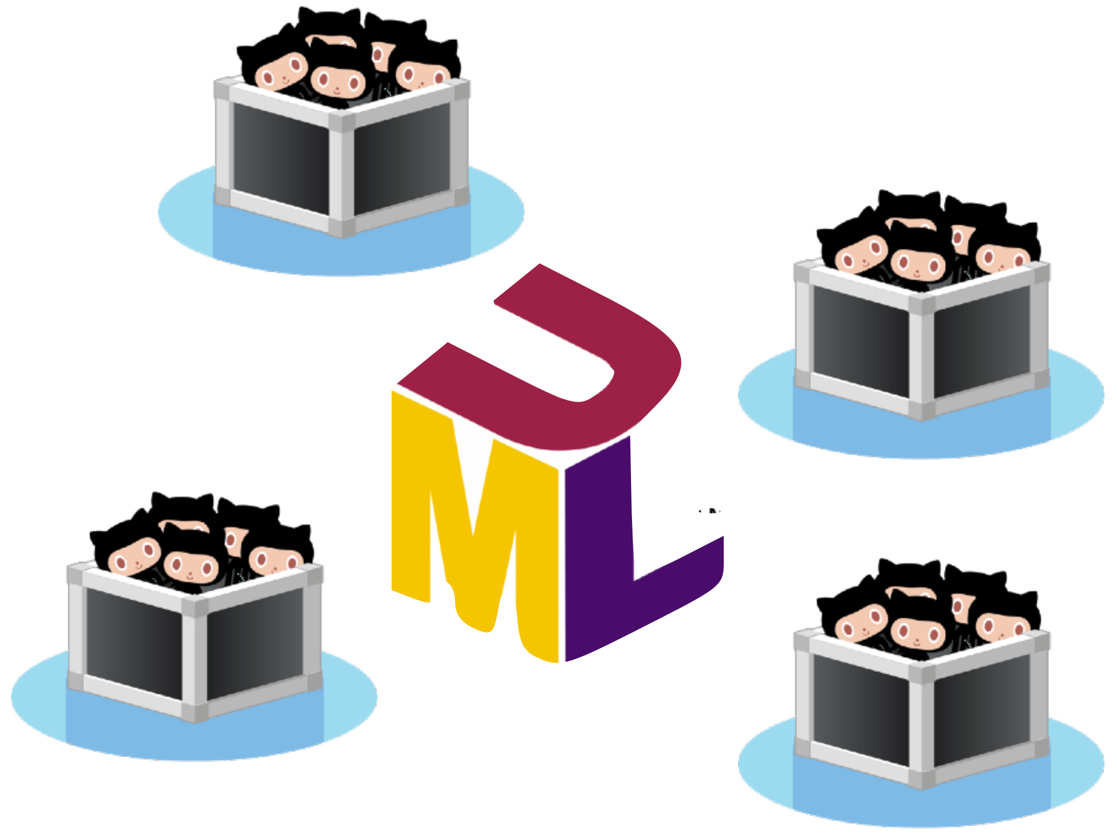
"Control" group



93 projects

Same filters, sampled using GHTorrent

"Treatment" group

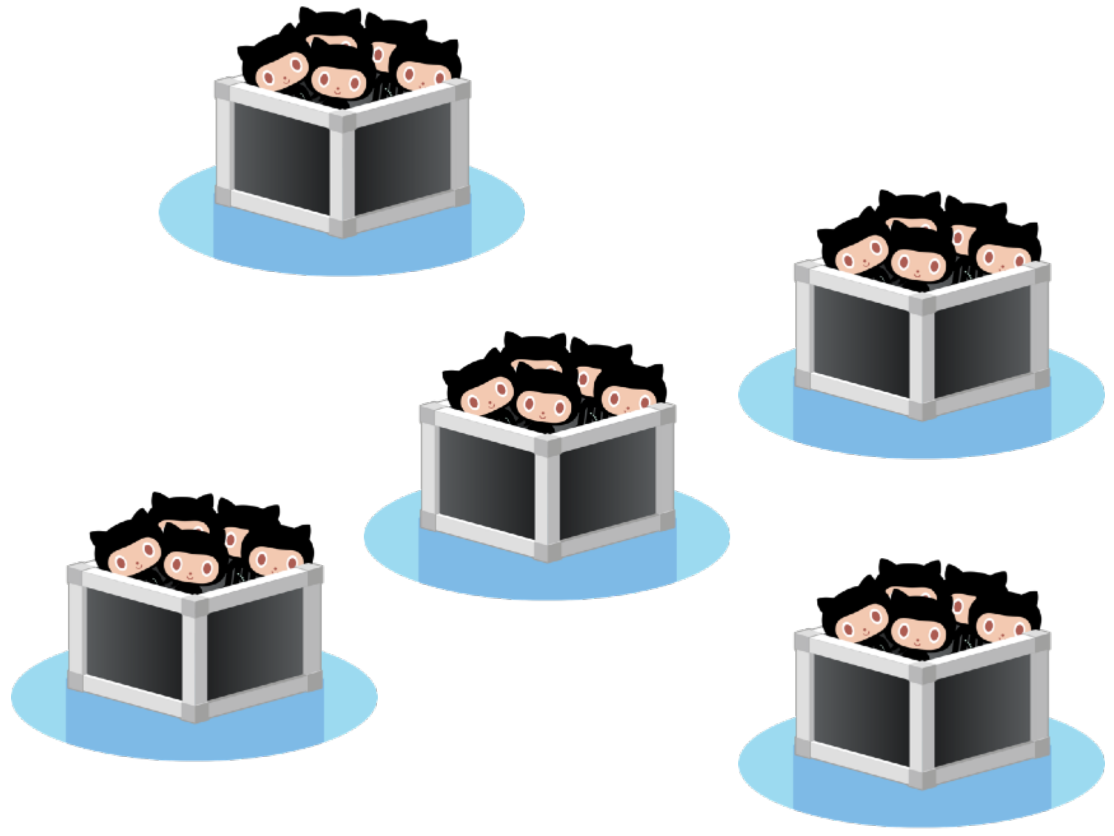


50 projects

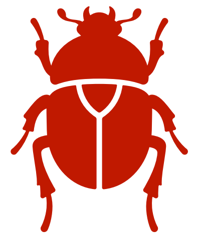
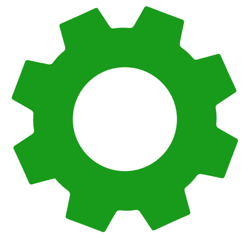
- C++, C#, Java
- 2009+
- 10+ stars
- 30+ issues
- Issues in English

Natural experiment

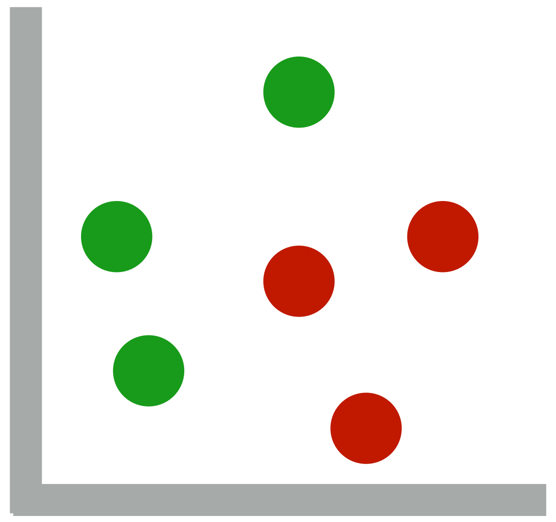
"Control" group



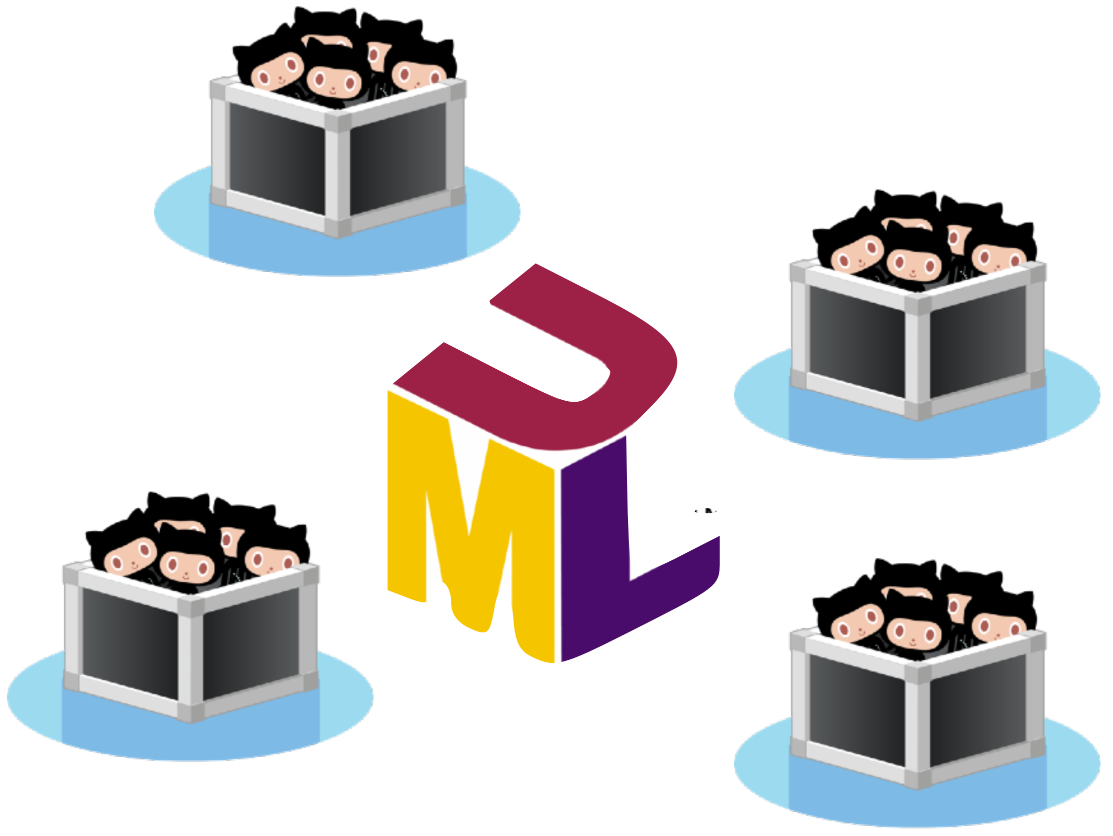
Separate bugs from features etc.

 vs  …

Naive Bayes classifier
89% accuracy

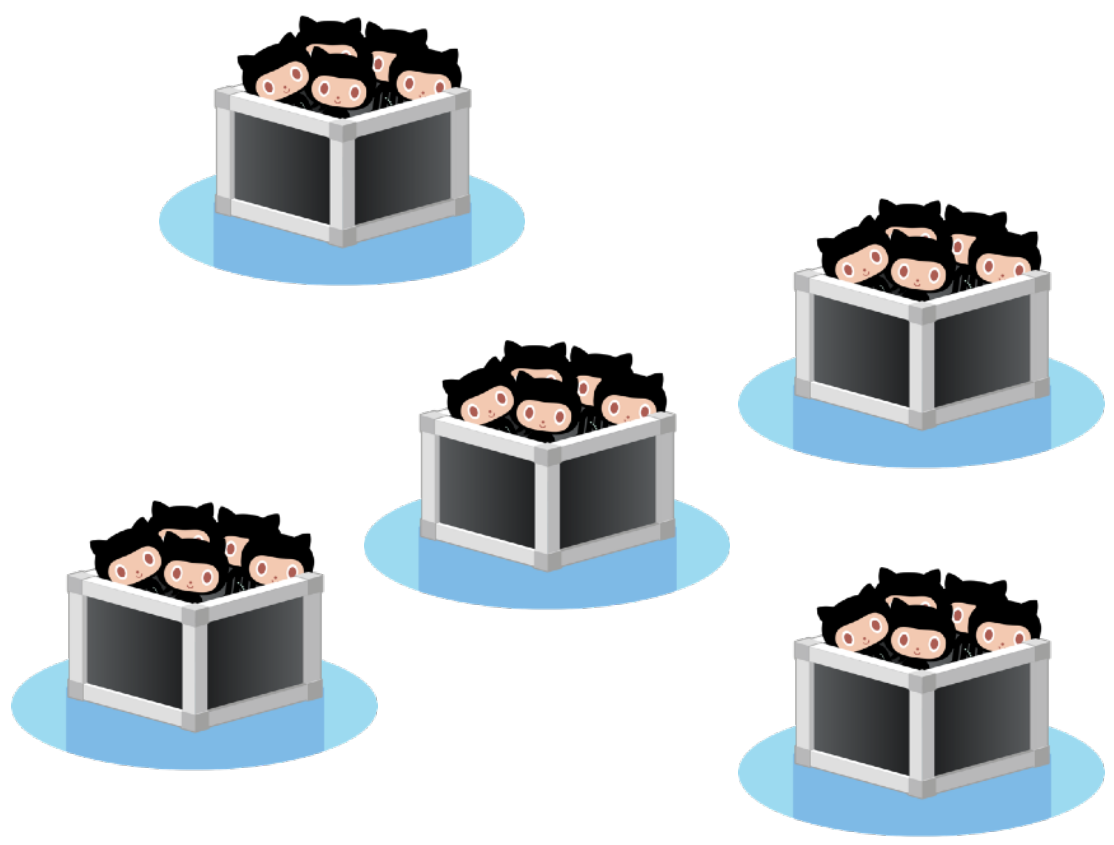


"Treatment" group

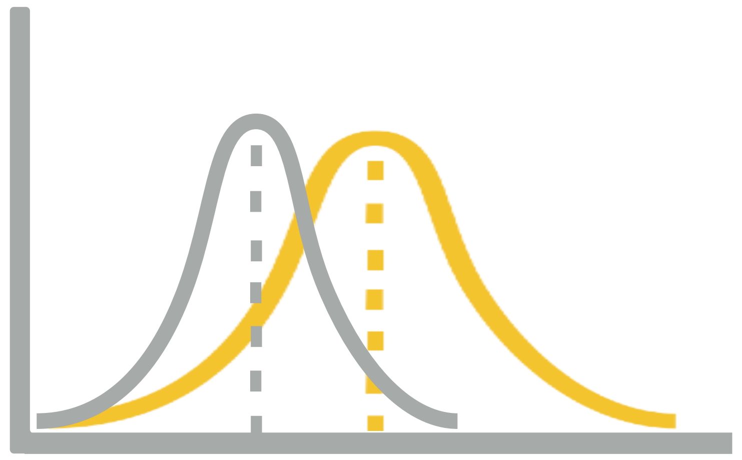


Natural experiment

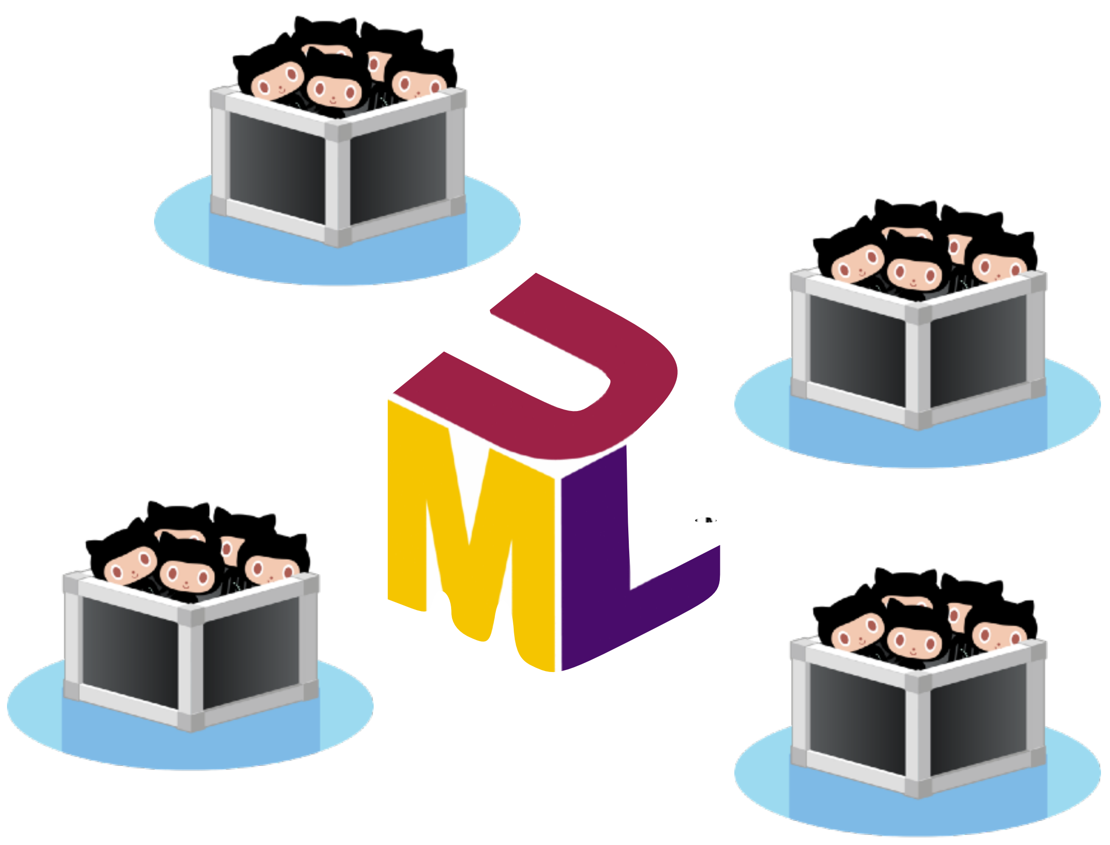
“Control” group



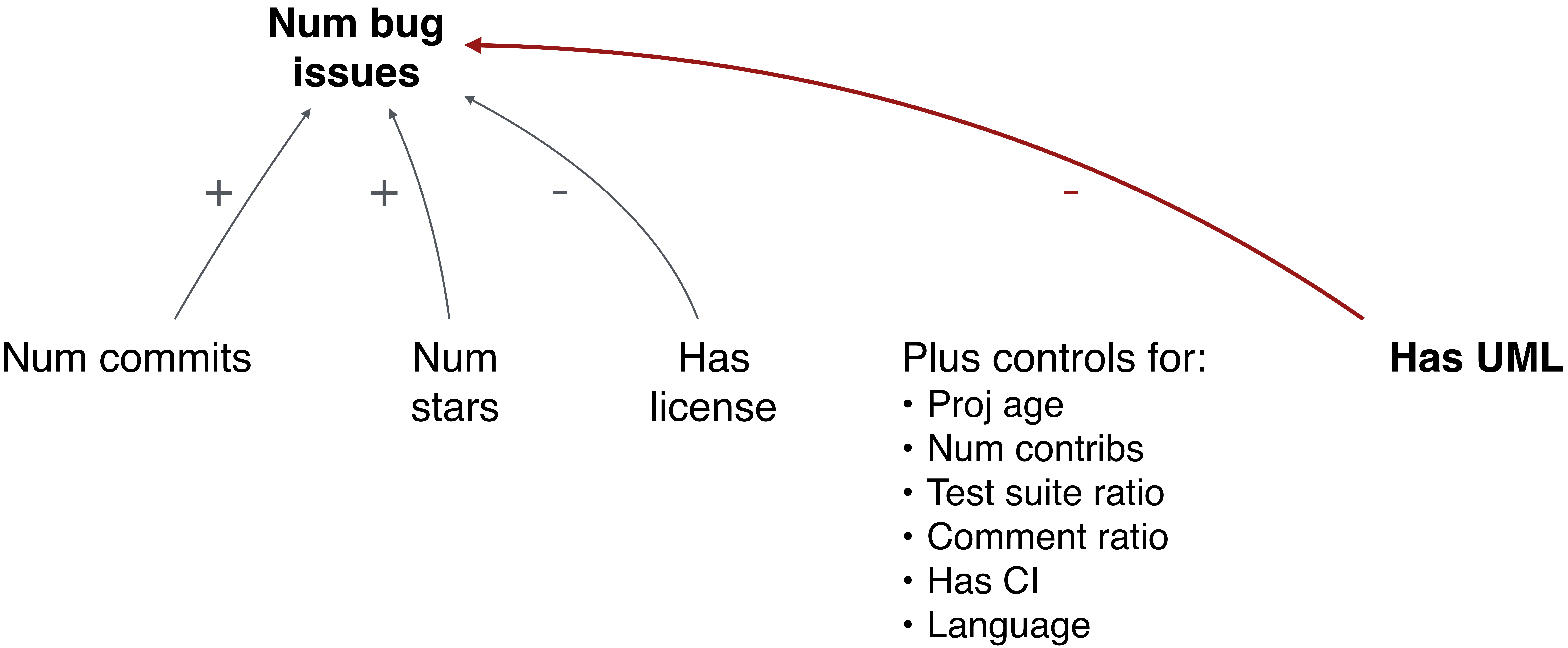
Compare
defect rates



“Treatment” group



Projects w/ UML: ~35% fewer bugs reported than projects w/t UML (R^2 = 58%)



Projects w/ UML: ~35% fewer bugs reported than projects w/t UML (R^2 = 58%)

Num bug issues

\+ \+ - -

< 0.5% of GitHub repos have UML

Only 50 UML projects in this model

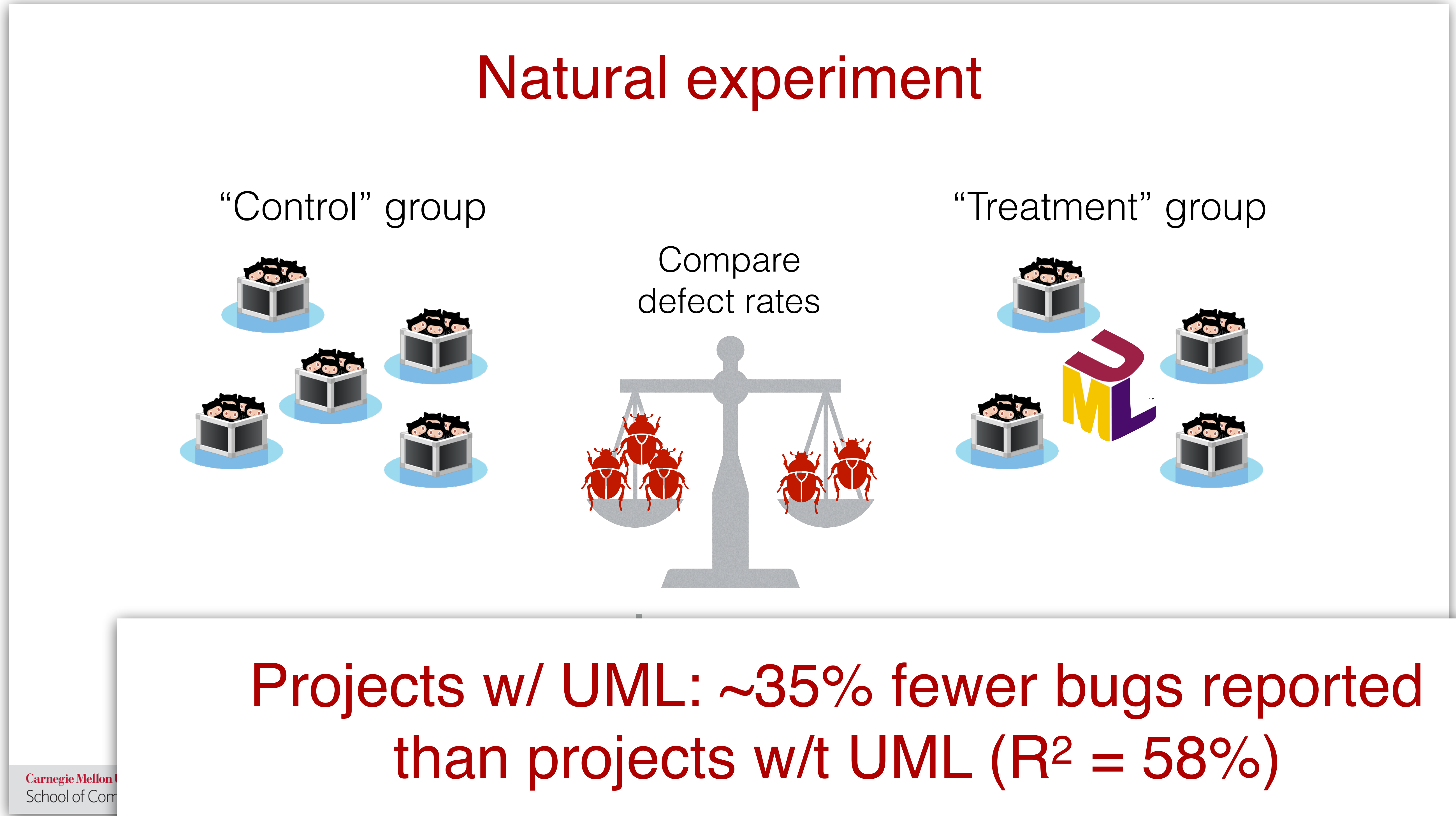
Num commits
(57% of variance explained)

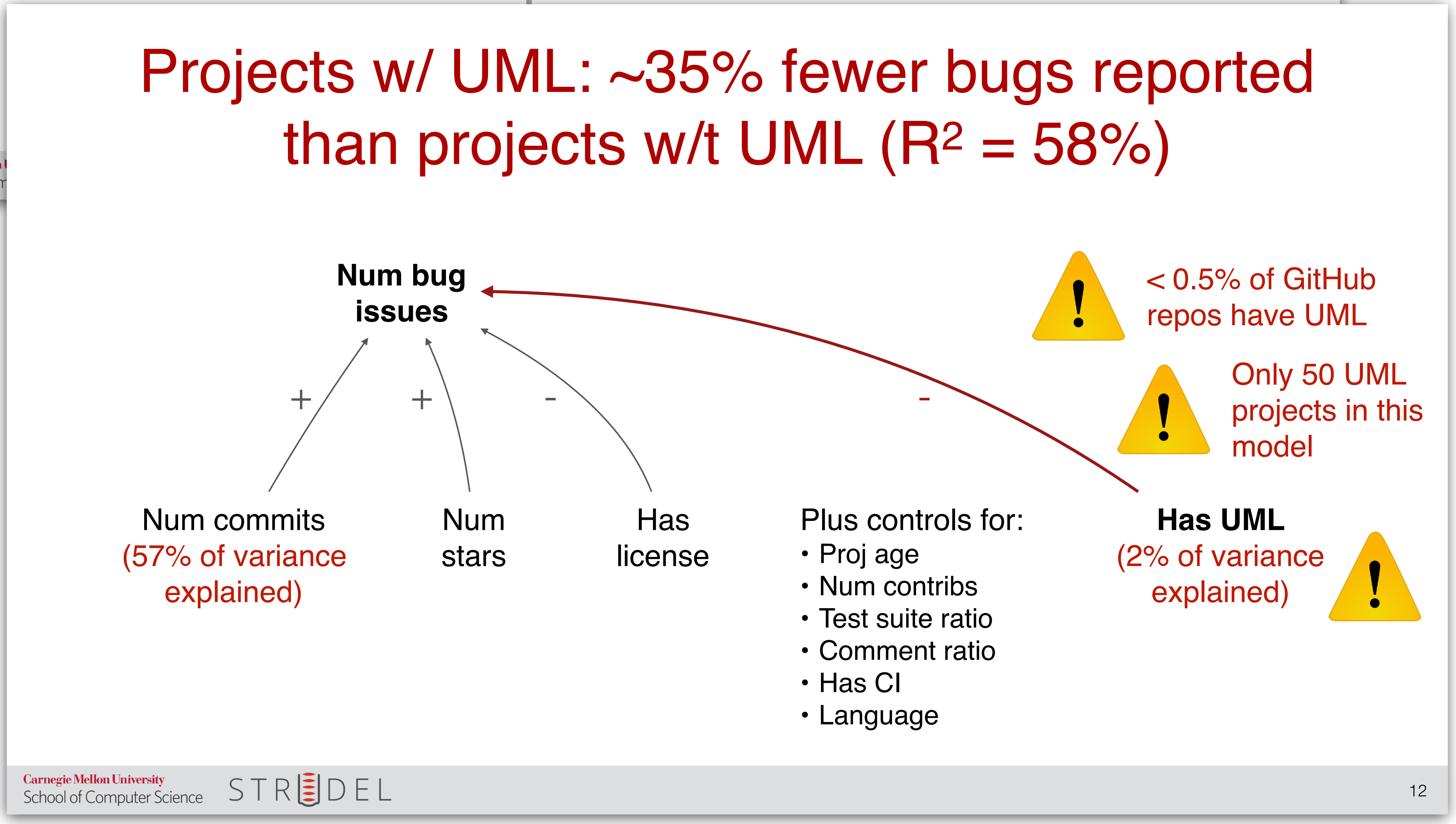
Num stars

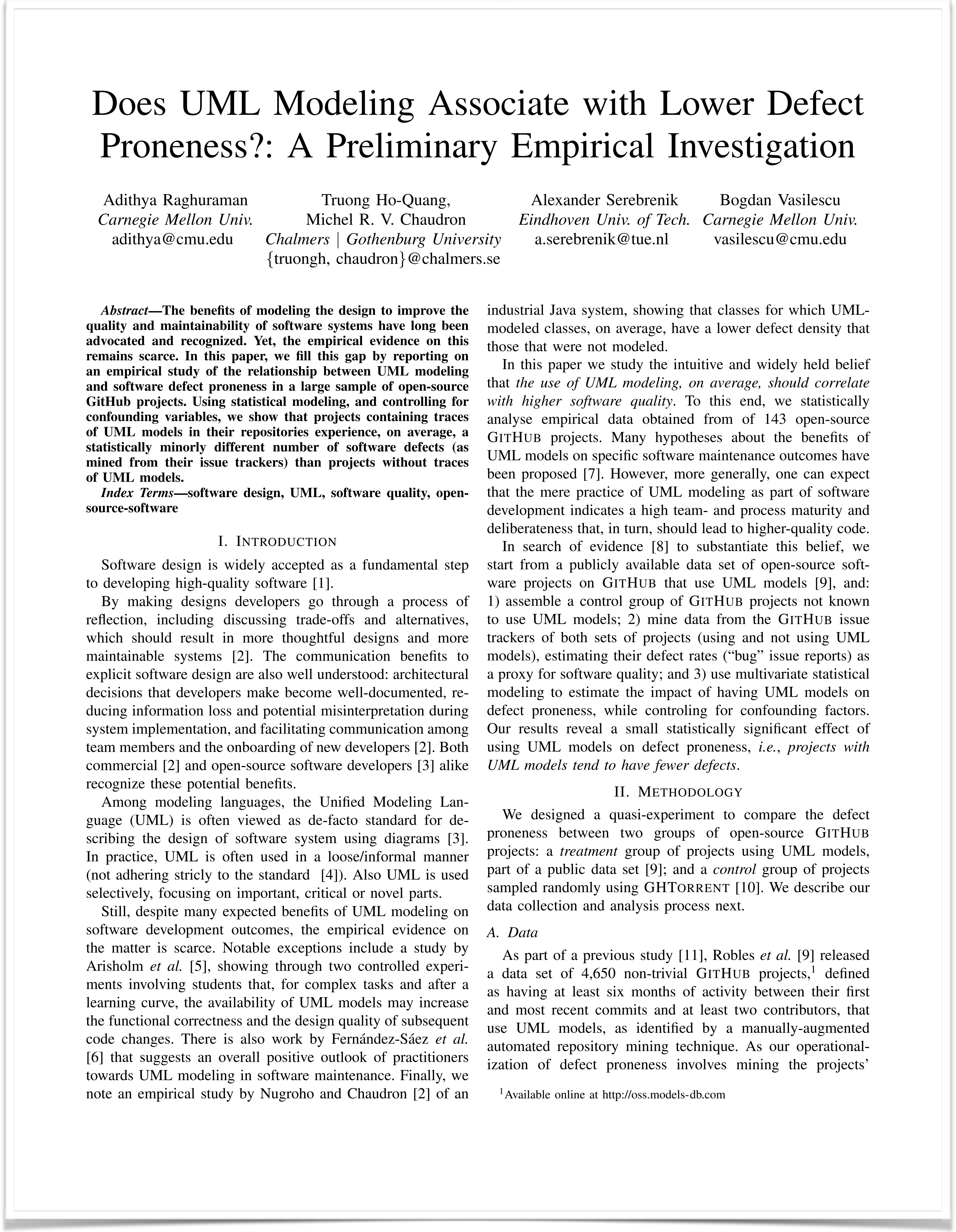
Has license

Plus controls for:
- Proj age
- Num contribs
- Test suite ratio
- Comment ratio
- Has CI
- Language

Has UML
(2% of variance explained)







Does UML Modeling Associate with Lower Defect Proneness?: A Preliminary Empirical Investigation

Adithya Raghuraman, *Carnegie Mellon Univ.*, adithya@cmu.edu
Truong Ho-Quang, Michel R. V. Chaudron, *Chalmers | Gothenburg University*, {truongh, chaudron}@chalmers.se
Alexander Serebrenik, *Eindhoven Univ. of Tech.*, a.serebrenik@tue.nl
Bogdan Vasilescu, *Carnegie Mellon Univ.*, vasilescu@cmu.edu

***Abstract*—The benefits of modeling the design to improve the quality and maintainability of software systems have long been advocated and recognized. Yet, the empirical evidence on this remains scarce. In this paper, we fill this gap by reporting on an empirical study of the relationship between UML modeling and software defect proneness in a large sample of open-source GitHub projects. Using statistical modeling, and controlling for confounding variables, we show that projects containing traces of UML models in their repositories experience, on average, a statistically minorly different number of software defects (as mined from their issue trackers) than projects without traces of UML models.**

***Index Terms*—software design, UML, software quality, open-source-software**

I. Introduction

Software design is widely accepted as a fundamental step to developing high-quality software [1].

By making designs developers go through a process of reflection, including discussing trade-offs and alternatives, which should result in more thoughtful designs and more maintainable systems [2]. The communication benefits to explicit software design are also well understood: architectural decisions that developers make become well-documented, reducing information loss and potential misinterpretation during system implementation, and facilitating communication among team members and the onboarding of new developers [2]. Both commercial [2] and open-source software developers [3] alike recognize these potential benefits.

Among modeling languages, the Unified Modeling Language (UML) is often viewed as de-facto standard for describing the design of software system using diagrams [3]. In practice, UML is often used in a loose/informal manner (not adhering stricly to the standard [4]). Also UML is used selectively, focusing on important, critical or novel parts.

Still, despite many expected benefits of UML modeling on software development outcomes, the empirical evidence on the matter is scarce. Notable exceptions include a study by Arisholm *et al.* [5], showing through two controlled experiments involving students that, for complex tasks and after a learning curve, the availability of UML models may increase the functional correctness and the design quality of subsequent code changes. There is also work by Fernández-Sáez *et al.* [6] that suggests an overall positive outlook of practitioners towards UML modeling in software maintenance. Finally, we note an empirical study by Nugroho and Chaudron [2] of an industrial Java system, showing that classes for which UML-modeled classes, on average, have a lower defect density that those that were not modeled.

In this paper we study the intuitive and widely held belief that *the use of UML modeling, on average, should correlate with higher software quality*. To this end, we statistically analyse empirical data obtained from of 143 open-source GitHub projects. Many hypotheses about the benefits of UML models on specific software maintenance outcomes have been proposed [7]. However, more generally, one can expect that the mere practice of UML modeling as part of software development indicates a high team- and process maturity and deliberateness that, in turn, should lead to higher-quality code.

In search of evidence [8] to substantiate this belief, we start from a publicly available data set of open-source software projects on GitHub that use UML models [9], and: 1) assemble a control group of GitHub projects not known to use UML models; 2) mine data from the GitHub issue trackers of both sets of projects (using and not using UML models), estimating their defect rates (“bug” issue reports) as a proxy for software quality; and 3) use multivariate statistical modeling to estimate the impact of having UML models on defect proneness, while controling for confounding factors. Our results reveal a small statistically significant effect of using UML models on defect proneness, *i.e.*, *projects with UML models tend to have fewer defects*.

II. Methodology

We designed a quasi-experiment to compare the defect proneness between two groups of open-source GitHub projects: a *treatment* group of projects using UML models, part of a public data set [9]; and a *control* group of projects sampled randomly using GHTorrent [10]. We describe our data collection and analysis process next.

A. Data

As part of a previous study [11], Robles *et al.* [9] released a data set of 4,650 non-trivial GitHub projects,[1] defined as having at least six months of activity between their first and most recent commits and at least two contributors, that use UML models, as identified by a manually-augmented automated repository mining technique. As our operationalization of defect proneness involves mining the projects’

[1] Available online at http://oss.models-db.com

Bogdan Vasilescu
@b_vasilescu
http://cmustrudel.github.io