Sharing Specifications

Christian Collberg Todd Proebsting

University of Arizona

The Opening Gambit

The Study

The Proposal

[redacted]

Abstract

We present a new general technique for protecting clients in distributed systems against *Remote Man-at-the-end* (R-MATE) attacks. Such attacks occur in settings where an adversary has physical access to an untrusted client device and can obtain an advantage from tampering with the hardware itself or the software it contains.

In our system, the trusted server overwhelms the untrusted client's analytical abilities by continuously and automatically generating and pushing to him diverse client code variants. The diversity subsystem employs a set of primitive code transformations that provide an ever-changing attack target for the adversary, making tampering difficult without this being detected by the server.

1. Introduction

Man-at-the-end (MATE) attacks occur in settings where an adversary has physical access to a device and compromises it by tampering with its hardware or software. *Remote* man-at-the-end (R-MATE) attacks occur in distributed systems where *untrusted clients* are in frequent communication with *trusted servers* over a network, and malicious user can get an advantage by compromising an untrusted device.

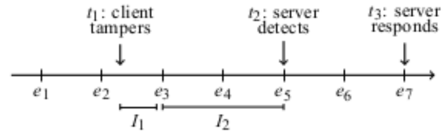
To illustrate the ubiquity of R-MATE vulnerabilities, consider the following four scenarios. First, in the *Advanced Metering Infrastructure* (AMI) for controlling the electrical power grid, networked devices ("*smart meters*") are installed at individual house-holds to allow two-way communication with control servers of the utility company. In an R-MATE attack against the AMI, a malicious consumer tampers with the meter to emulate an imminent blackout, or to trick a control server to send disconnect commands to other customers [7, 21]. Second, massive multiplayer online games are susceptible to R-MATE attacks since a malicious player who tampers with the game client can get an advantage over other players [16]. Third, wireless sensors are often deployed in unsecured environments (such as theaters of war) where they are vulnerable to tampering attempts. A compromised sensor could be coached into supplying the wrong observations to a base station, causing real-world damage. Finally, while electronic health records (EHR) are typically protected by encryption while stored in databases and in transit to doctors' offices, they are vulnerable to R-MATE attack if an individual doctor's client machine is compromised.

1.1 Overview

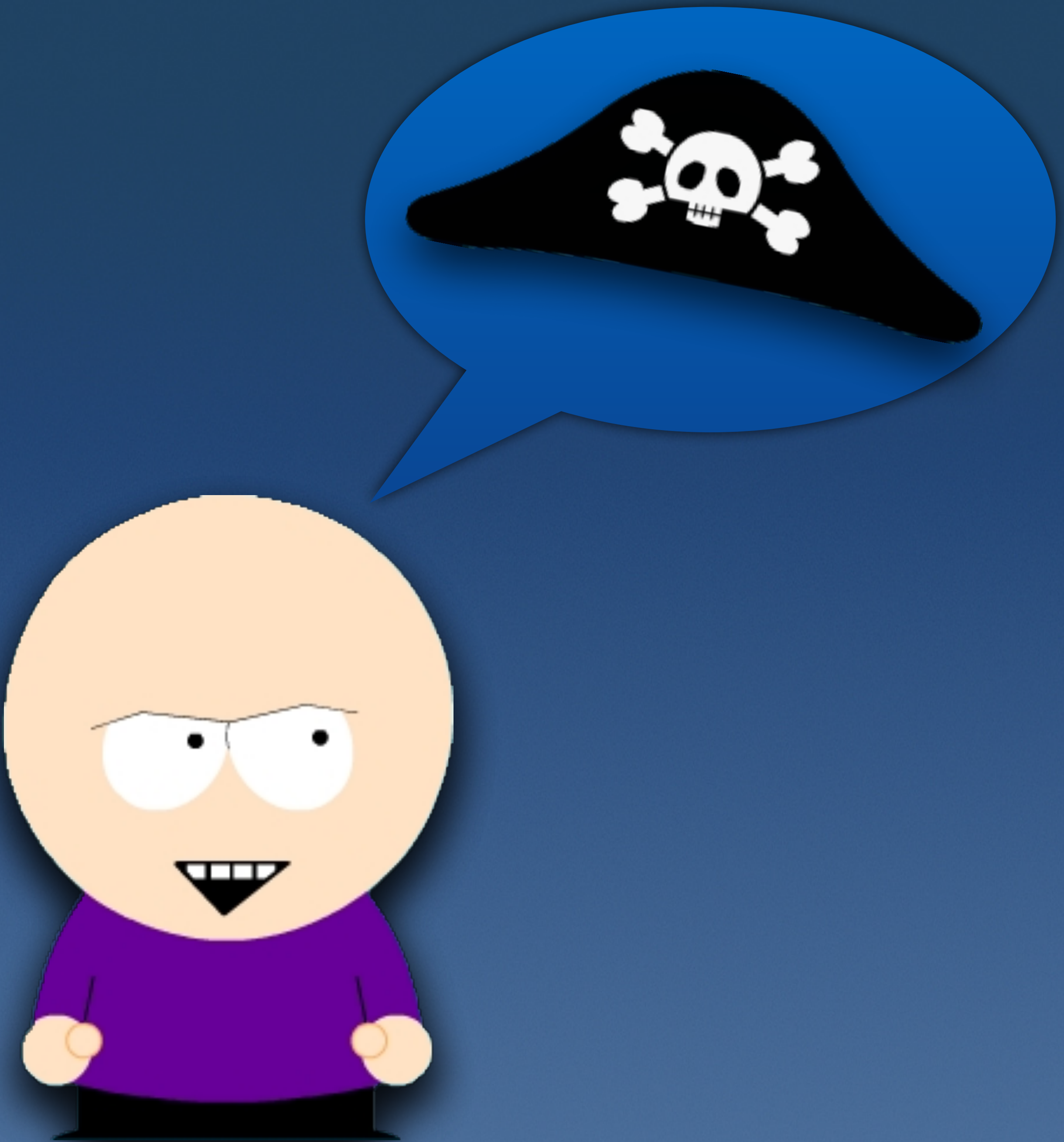
In each of the scenarios above the adversary's goal is to tamper with the client code and data under his control. The trusted server's goal is to *detect* any such integrity violations, after which countermeasures (such as severing connections, legal remedies, etc.) can be launched.

Security mechanisms. In this paper we present a system that achieves protection against R-MATE attacks through the extensive use of code diversity and continuous code replacement. In our system, the trusted server continuously and automatically generates diverse variants of client code, pushes these code updates to the untrusted clients, and installs them as the client is running. The intention is to force the client to constantly analyze and re-analyze incoming code variants, thereby overwhelming his analytical abilities, and making it difficult for him to tamper with the continuously changing code without this being detected by the trusted server.

Limitations. Our system specifically targets distributed applications which have frequent client-server communication, since client tampering can only be detected at client-server interaction events. Furthermore, while our use of code diversity can *delay* an attack, it cannot completely *prevent* it. Our goal is therefore the rapid *detection* of attacks; applications which need to completely prevent any tampering of client code, for even the shortest length of time, are not suitable targets for our system. To see this, consider the following timeline in the history of a distributed application running under our system:



The e_i:s are *interaction events*, points in time when clients communicate with servers either to exchange application data or to perform code updates. At time t_1 the client tampers with the code under his control. Until the next interaction event, during interval I_1, the client runs autonomously, and the server cannot detect the attack. At time t_2, after an interval I_2 consisting of a few interaction events, the client's tampering has caused it to display anomalous behavior, perhaps through the use of an outdated communication protocol, and the server detects this. At time t_3, finally, the server issues a response, perhaps by shutting




To: authors@cs.ux.edu
Hi again! We think our system can break yours! Can you please send us your code?

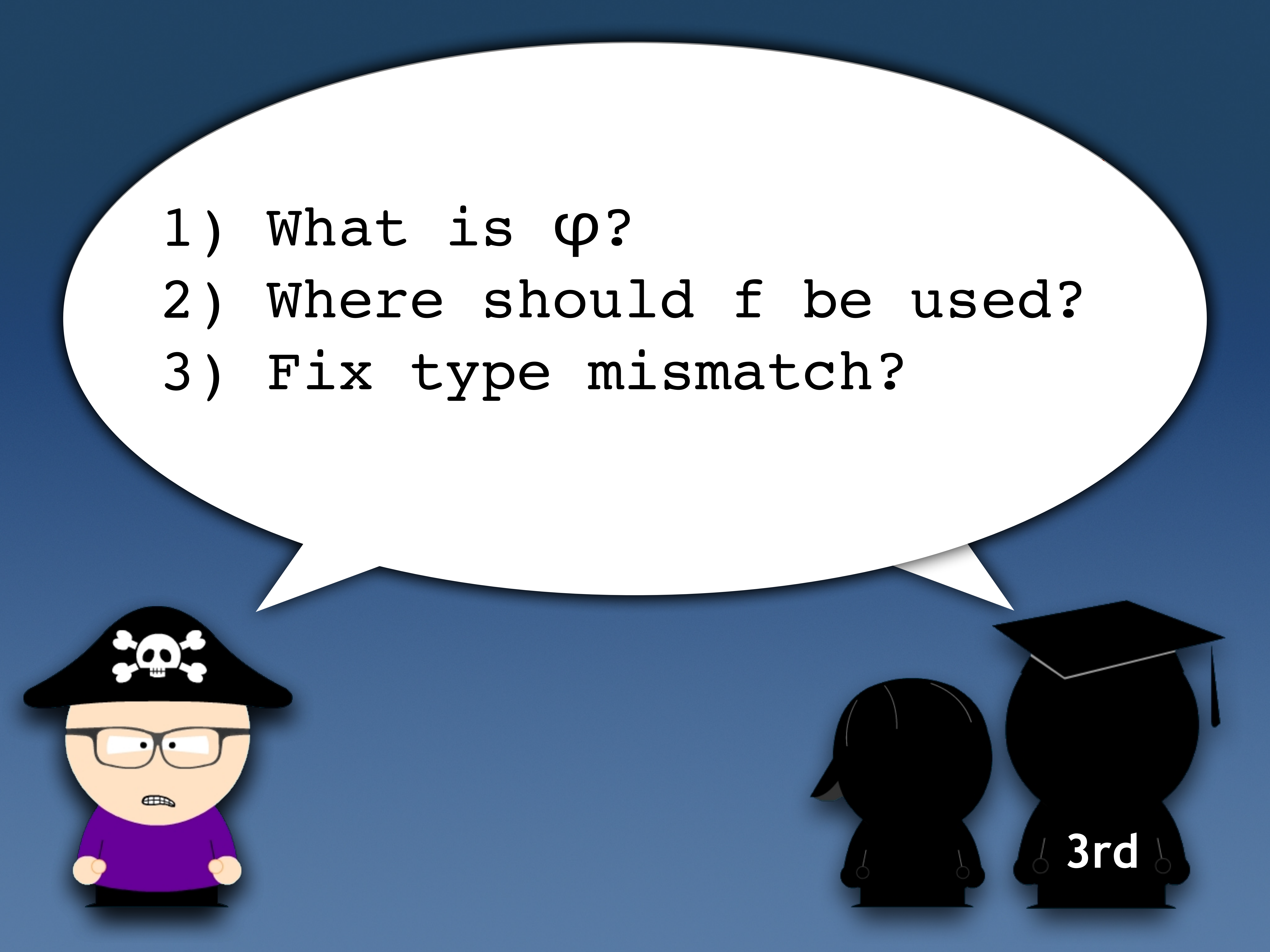
Haskell
A Purely Functional Language

Technical Report

Conference Paper

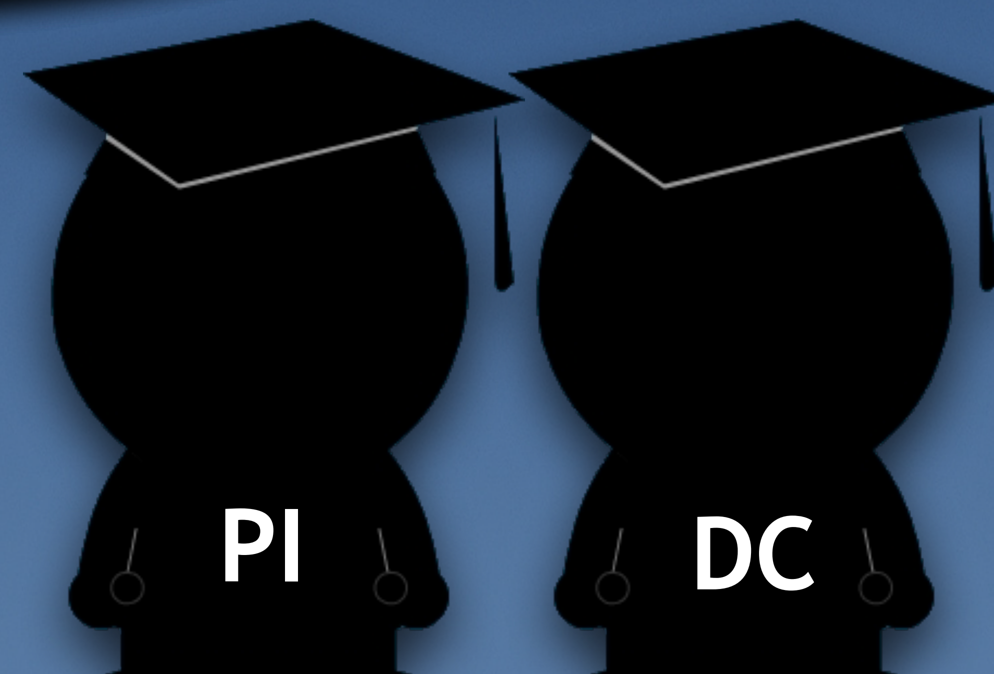
PhD Thesis

```
                    .
                    .
                    .
                    .

type operator =
   | NOP
   | LC of operand * value * binop
   | MR of operand * value * operand * binop
   | MW of operand * value * operand * binop
   | MV of operand * operand

                    .
                    .
                    .
                    .
```


1) What is φ?
2) Where should f be used?
3) Fix type mismatch?
3rd

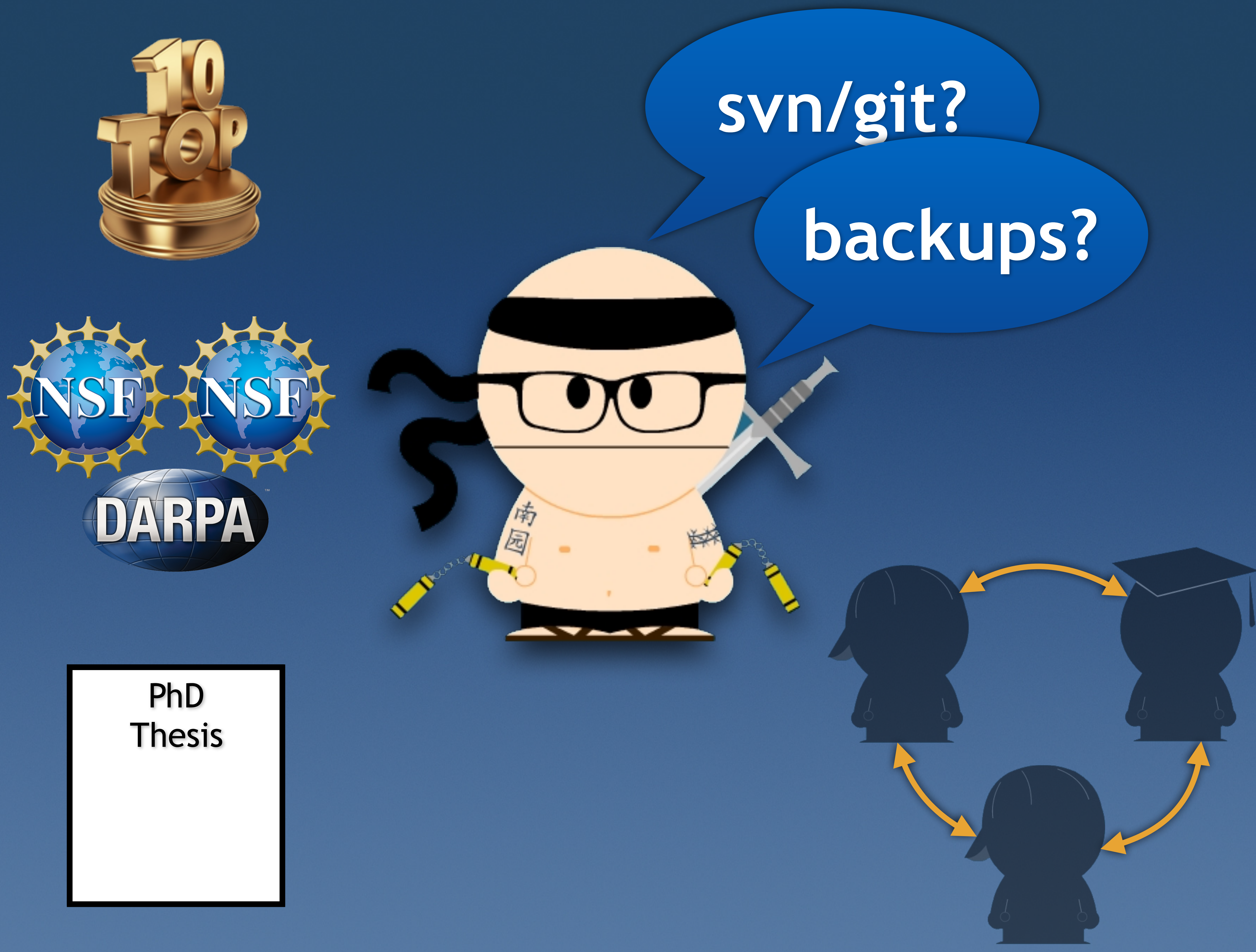
[T]he 2005 NSF Grant Policy Manual … states that

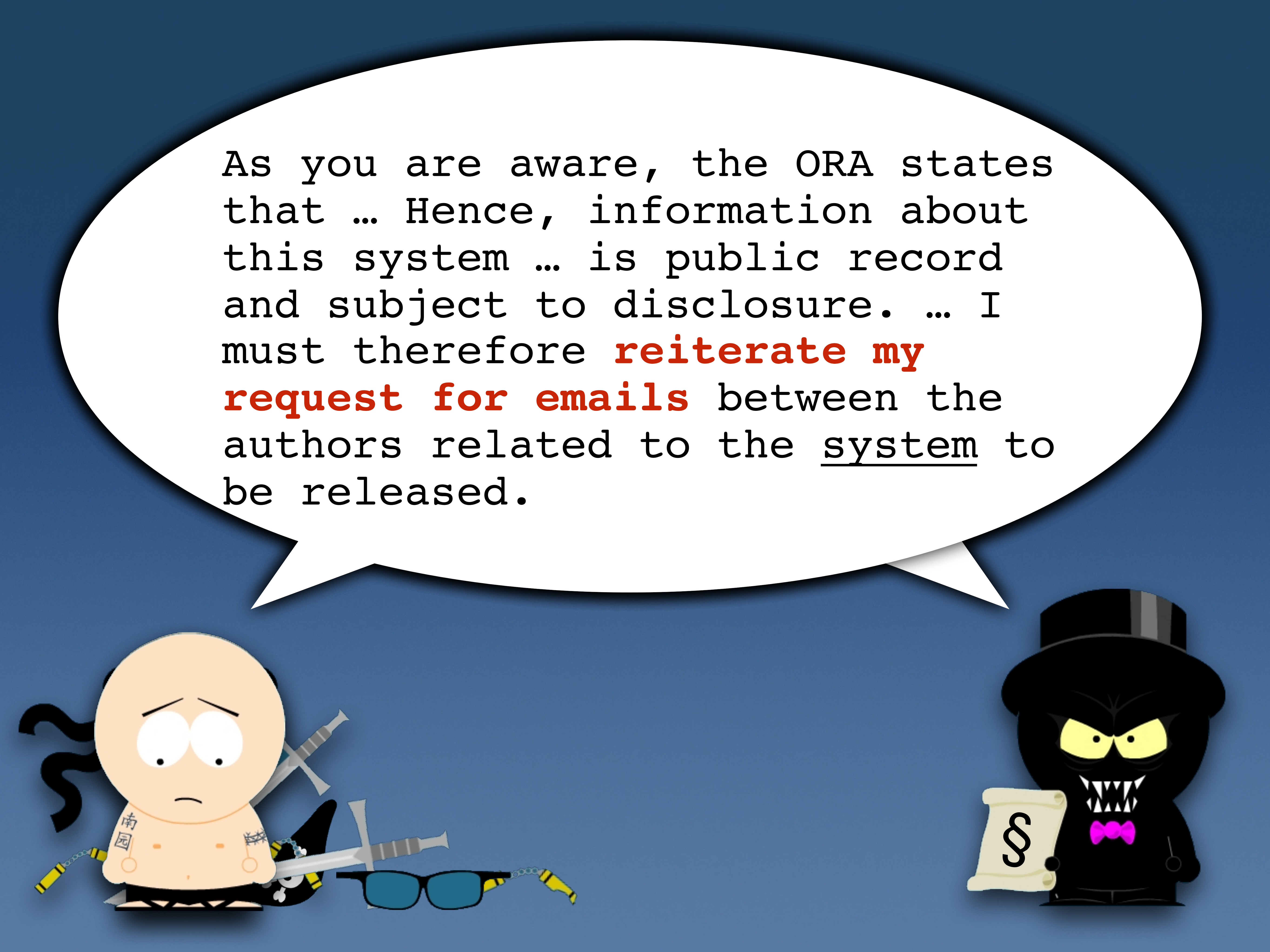
c) Investigators and grantees are **encouraged to share software** and inventions created under the grant or otherwise make them or their products widely available and usable.

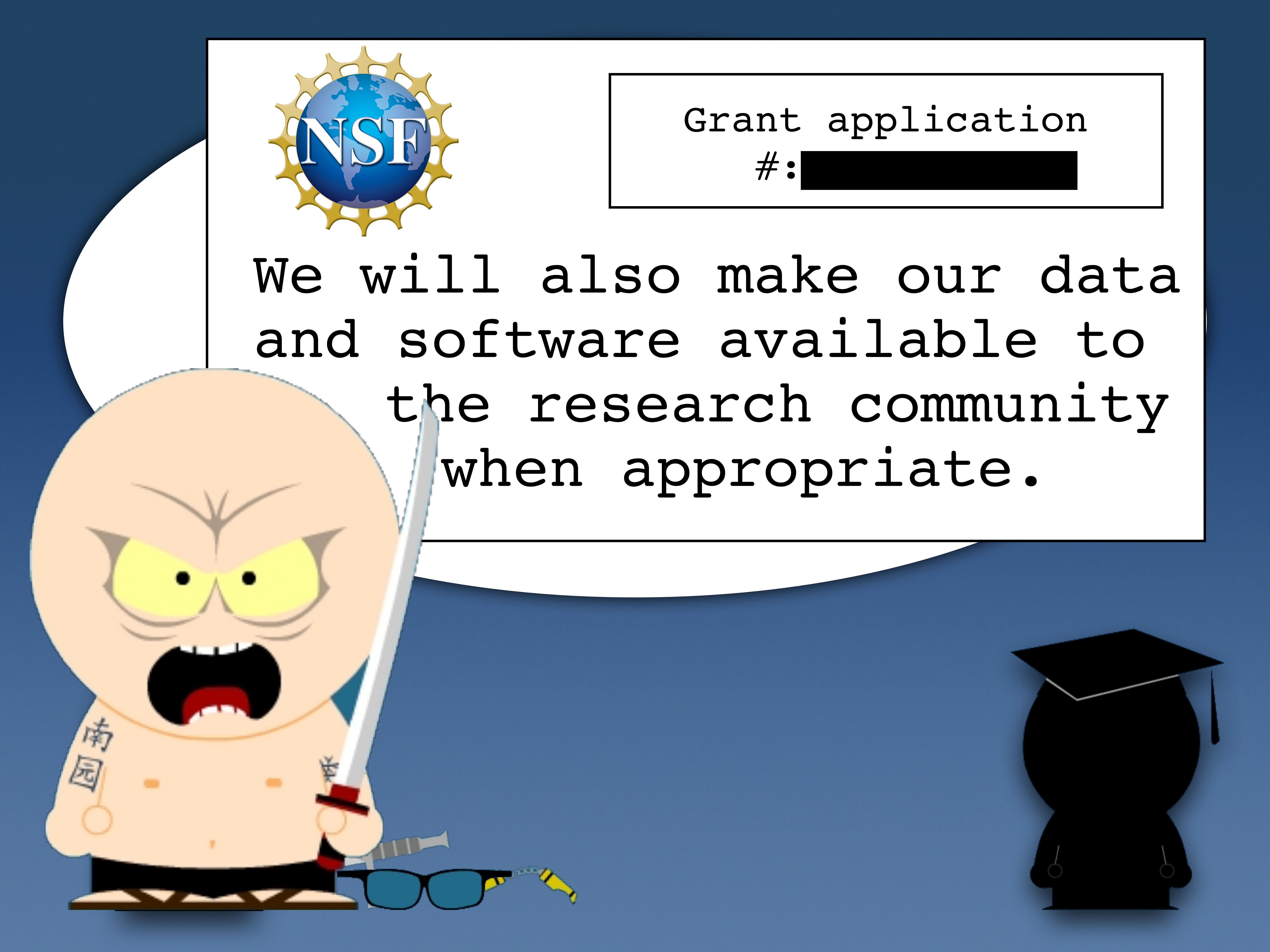


From: legal@cs.ux.edu

… to the extent such records may exist, **they will not be produced pursuant to ORA**.


svn/git?
backups?
NSF
NSF
DARPA
PhD
Thesis


As you are aware, the ORA states that … Hence, information about this system … is public record and subject to disclosure. … I must therefore reiterate my request for emails between the authors related to the system to be released.


NSF
Grant application
#: [redacted]
We will also make our data and software available to the research community when appropriate.
南园

Why?

Reasons for Sharing

Re

Rep

Be

So, why am I telling you this story? Well, what we learned from this story, is that people are not always willing to share.

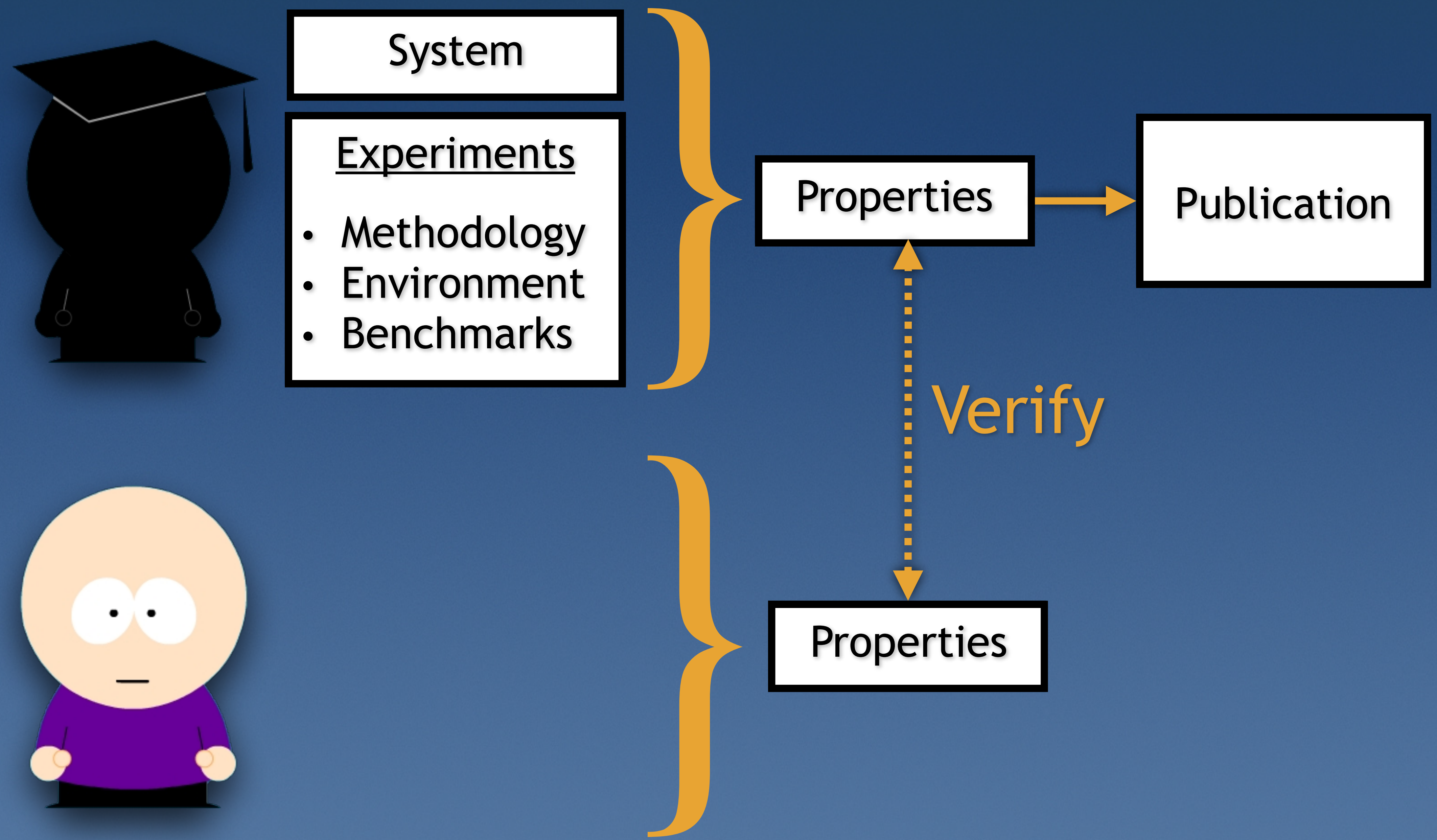
There are 3 reasons for why, as a researcher, you would want to share the artifacts that your research produces with other researchers.

First of all, you want the reviewers and readers of your work to trust it. In other words, it should be possible to repeat your experiments and get the same results. And if they don't have access to your code and data, how can they? That's what we call repeatability.

Second, it should be possible to independently verify the claims in your paper. That's what we call reproducibility. And, since in computer systems, we compress 100,000 lines of code into 15 pages of a research paper, completely understanding a work may require access to the code and data that went into it.

Finally, like your mother told you and your siblings, it is good to share. In science, it's good to share because it allows others to build on your work, and that will advance the field. And sharing for the good of scientific progress, we call benefaction.

Repeatability

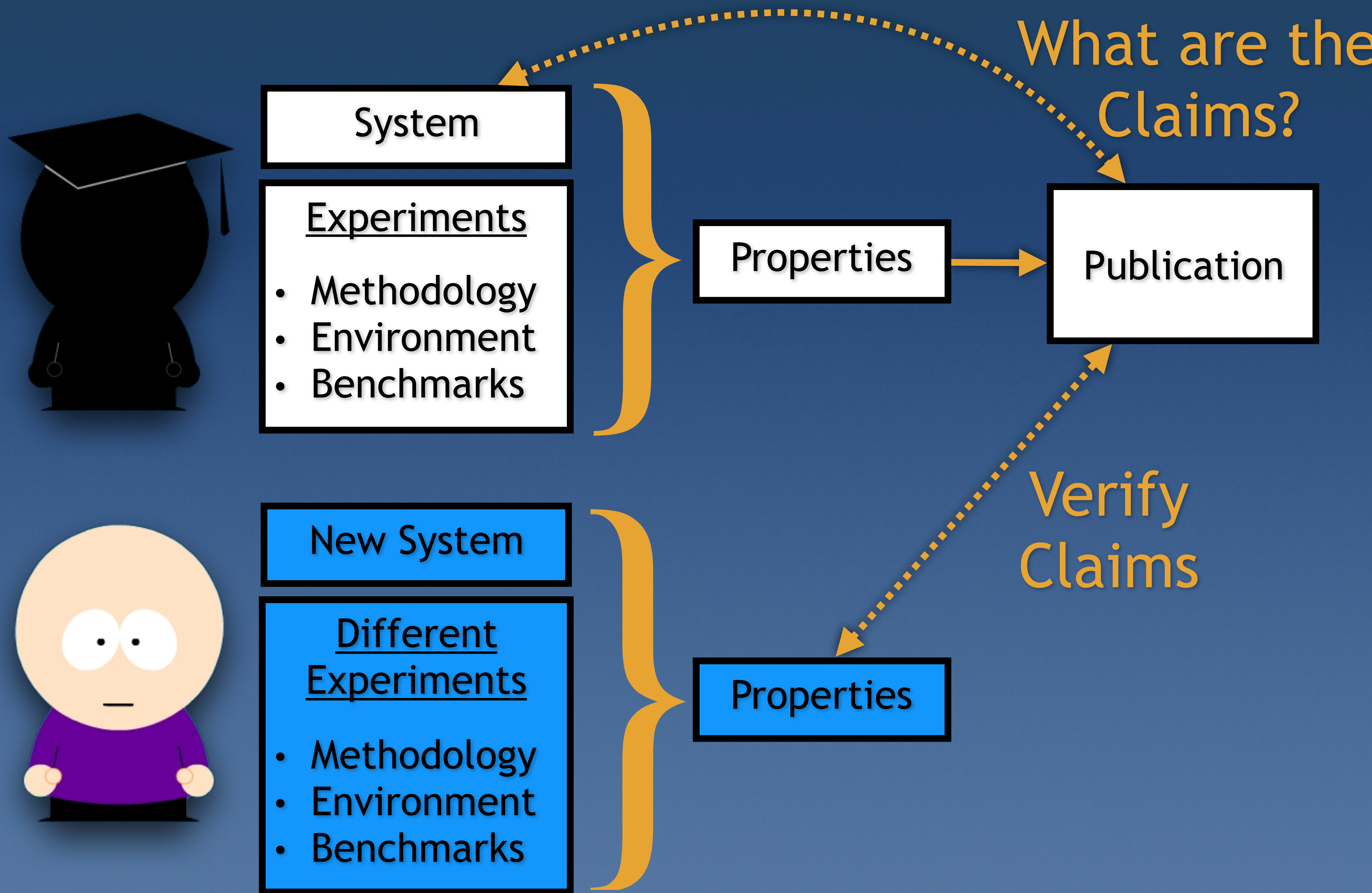


Repeatability

[T]he ability to re-run the exact same experiment with the same method on the same or similar system and obtain the same or very similar result.

Vitek, Kalibera: R3 – Repeatability, Reproducibility and Rigor

Reproducibility



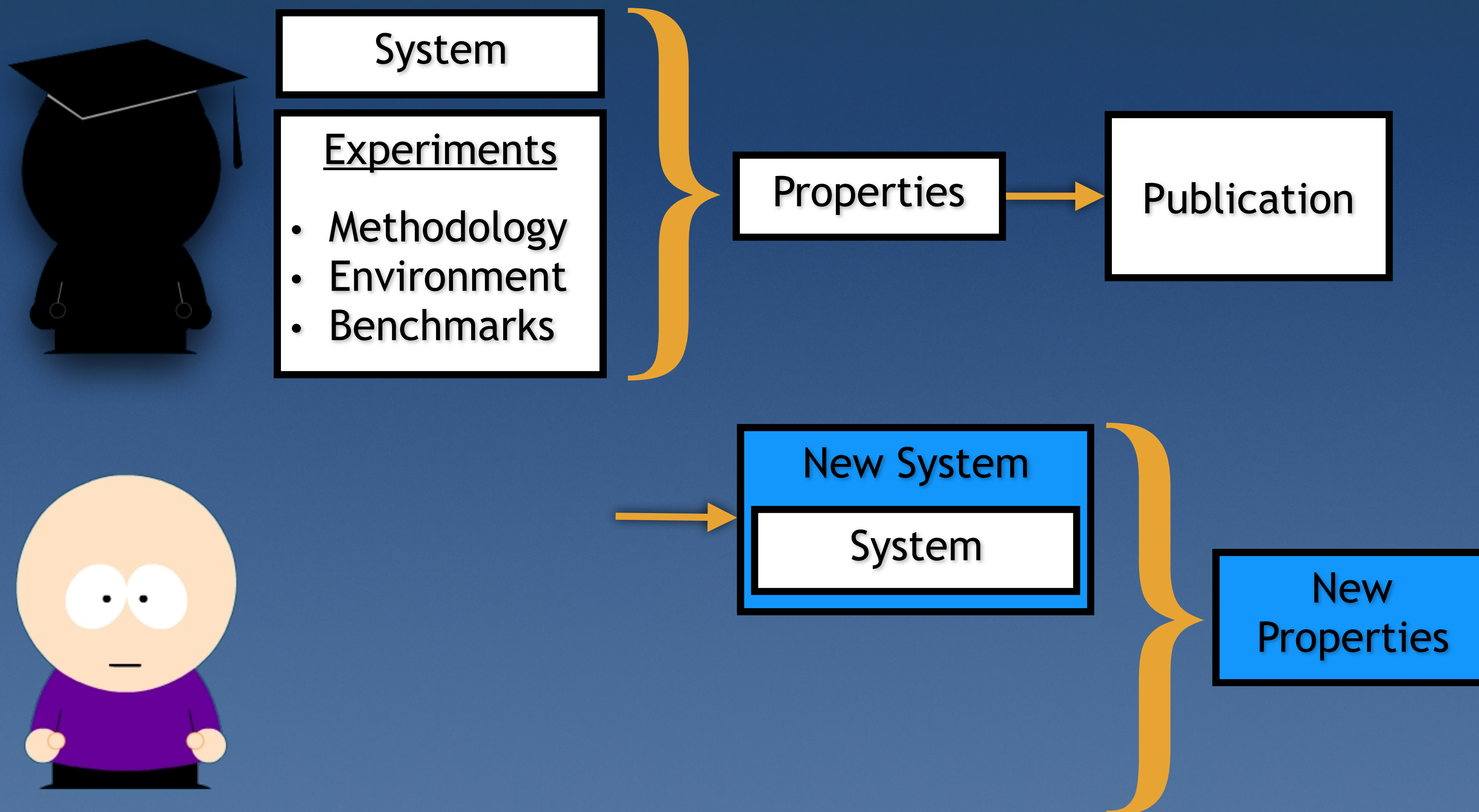
Reproducibility

[The] independent confirmation of a scientific hypothesis through reproduction by an independent researcher/lab…

[Is] carried out after a publication, based on the information in the paper…

Vitek, Kalibera: R3 – Repeatability, Reproducibility and Rigor

Benefaction



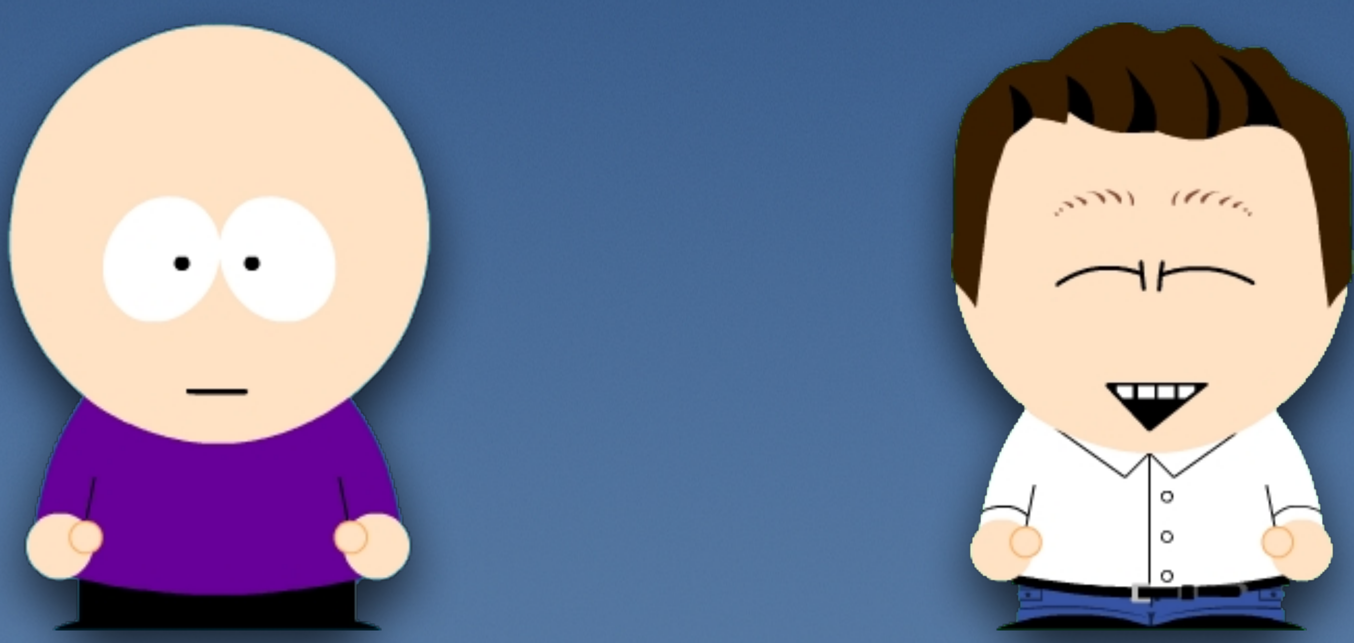
Benefaction

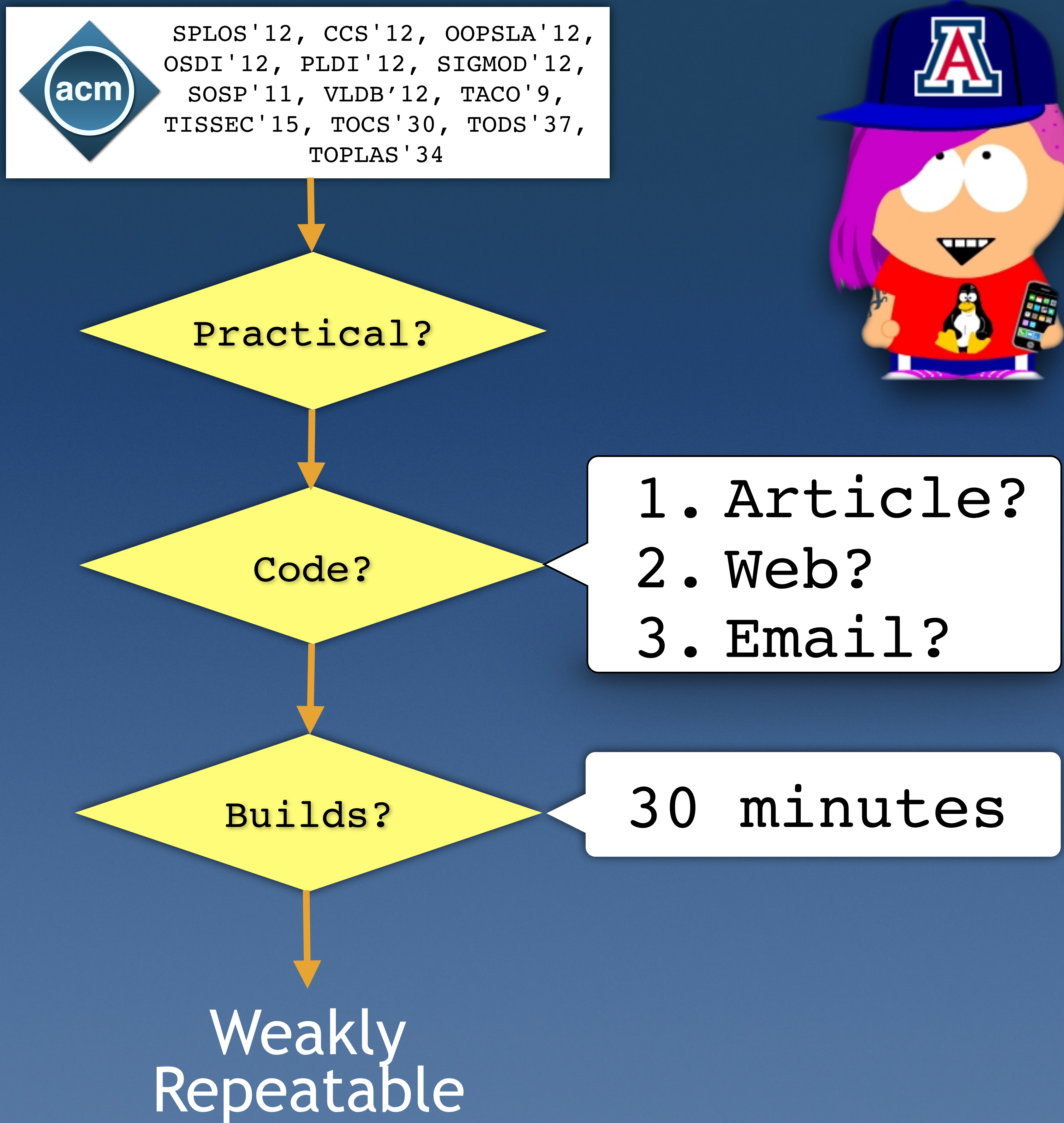
The avoidance of needless replication of work in order to better advance scientific progress.

The Study

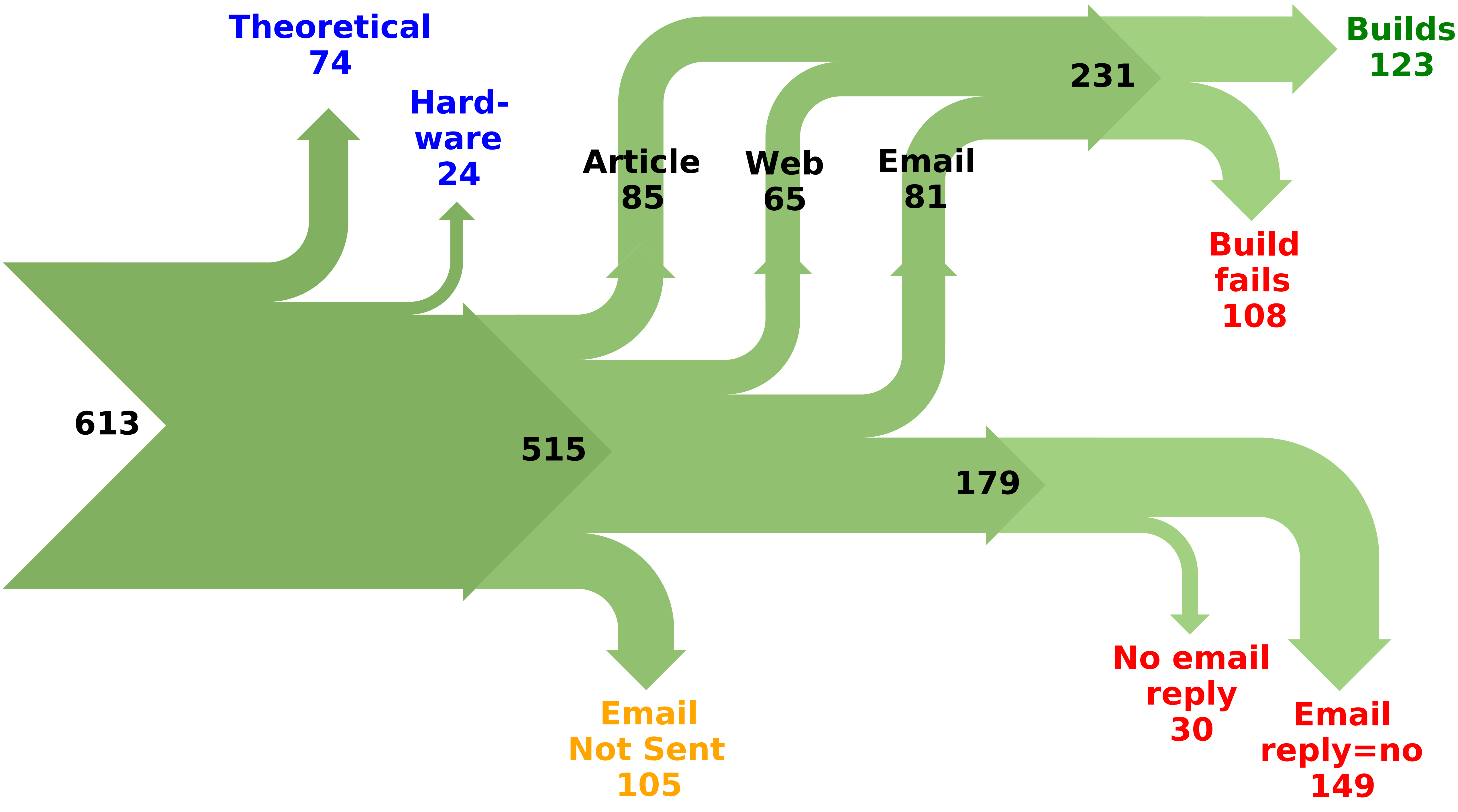
Weak Repeatability

Do authors make the source code used to create the results in their article available, and will it build?





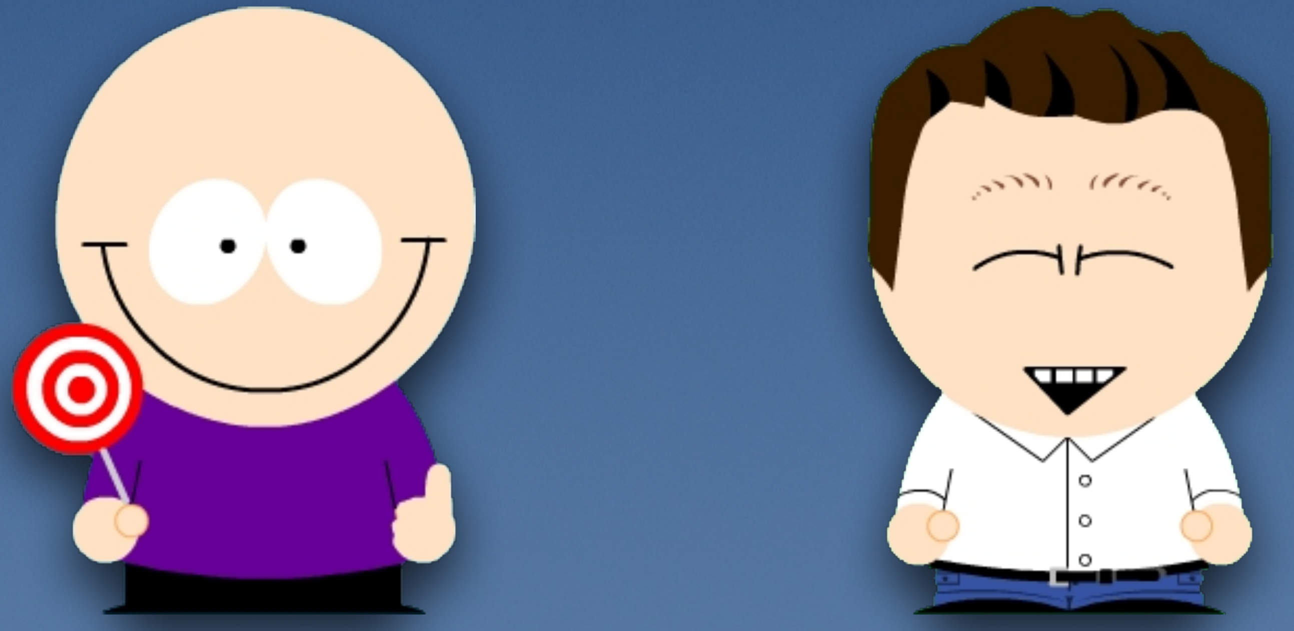
SPLOS'12, CCS'12, OOPSLA'12, OSDI'12, PLDI'12, SIGMOD'12, SOSP'11, VLDB'12, TACO'9, TISSEC'15, TOCS'30, TODS'37, TOPLAS'34
Practical?
Code?
1. Article?
2. Web?
3. Email?
Builds?
30 minutes
Weakly Repeatable



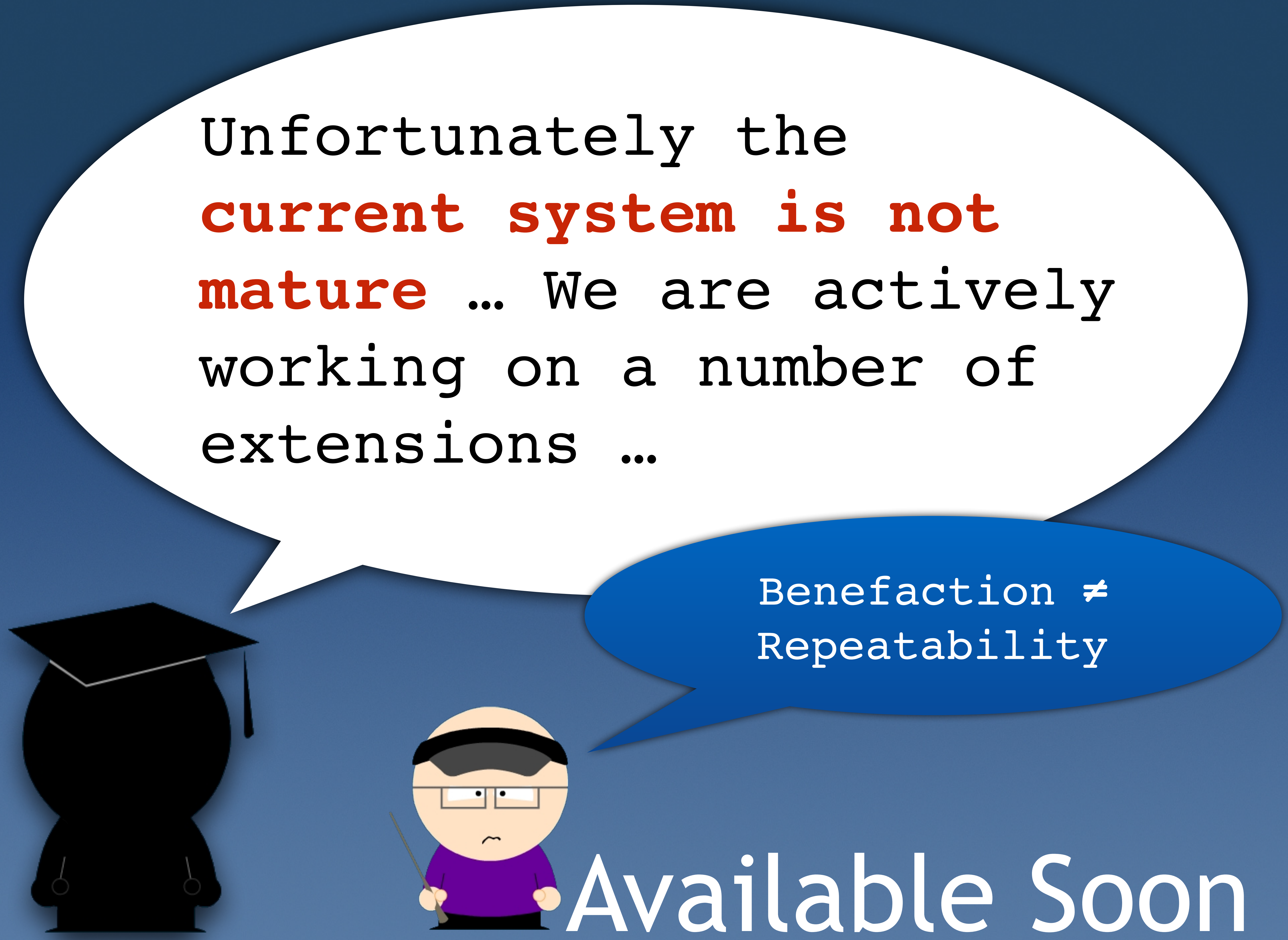
* Preliminary results - more about this later!

Reasons for not Sharing?

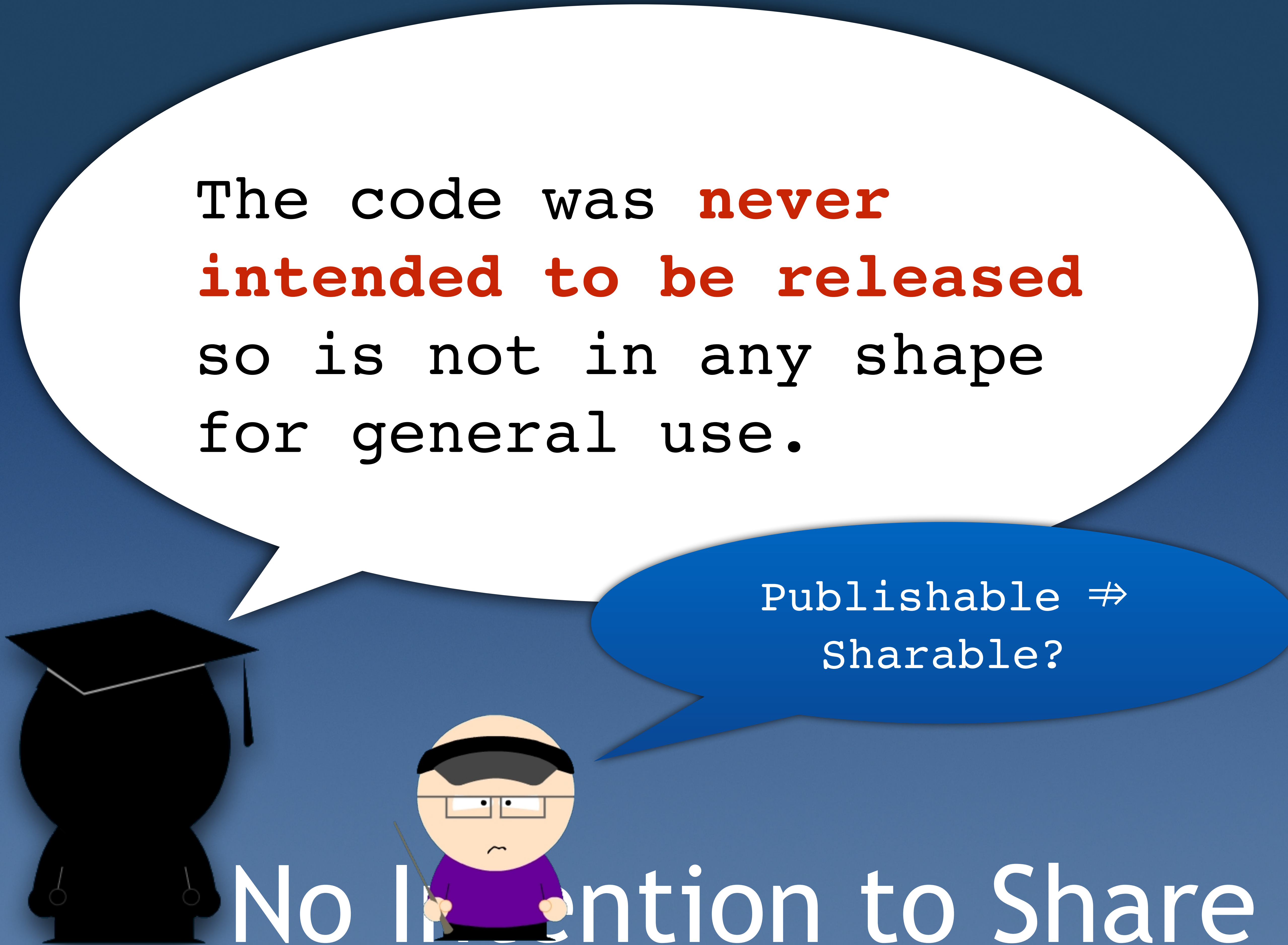
The email responses we received were pleasant, accommodating, and apologetic if code could not be provided.




The good news … I was able to find some code. I am just **hoping** that it … **matches the implementation** we … used for the paper.
git tag -a pldi-final
Versioning


Unfortunately the **current system is not mature** … We are actively working on a number of extensions …
Benefaction ≠ Repeatability
Available Soon

No Intention to Share

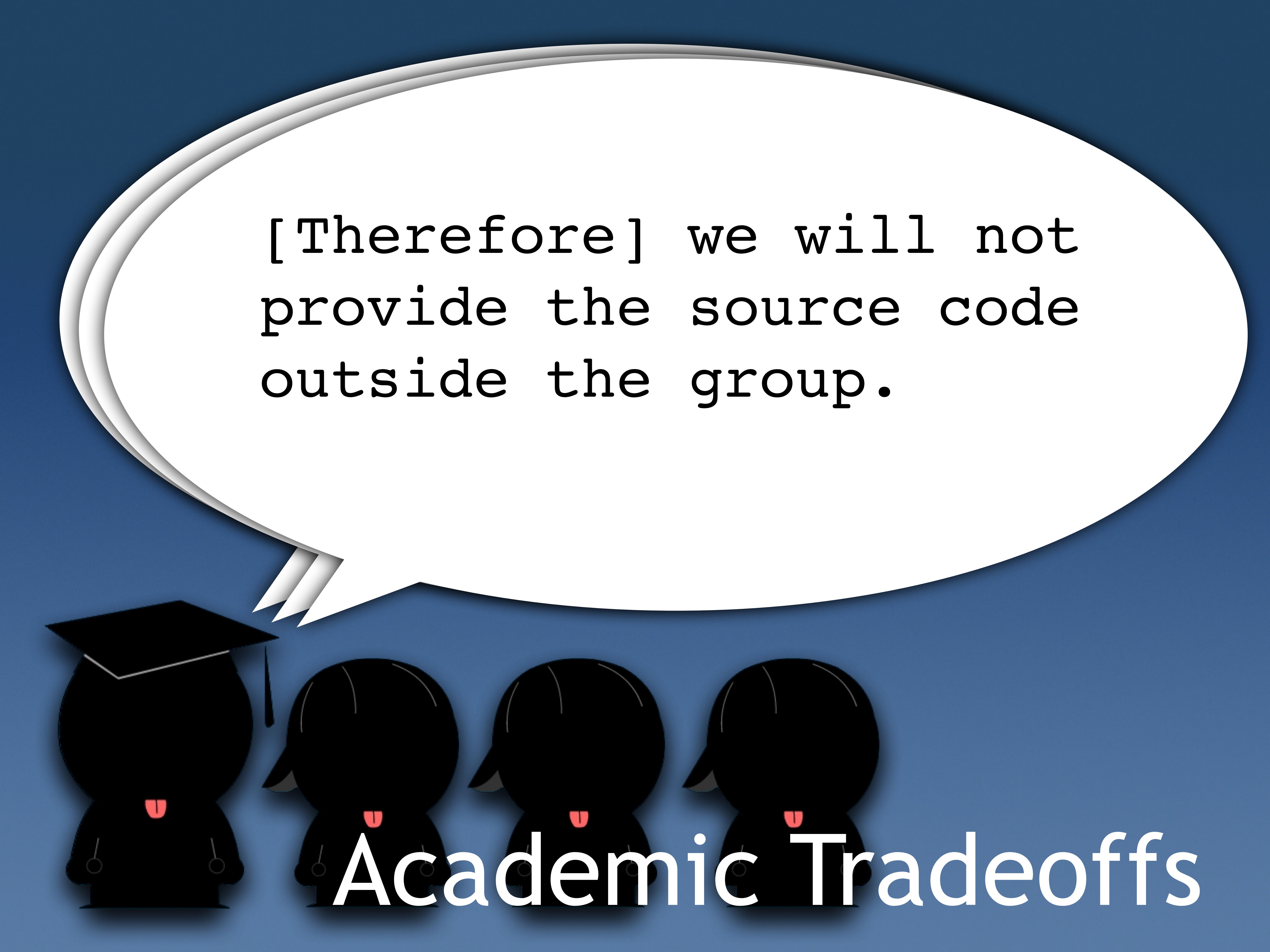


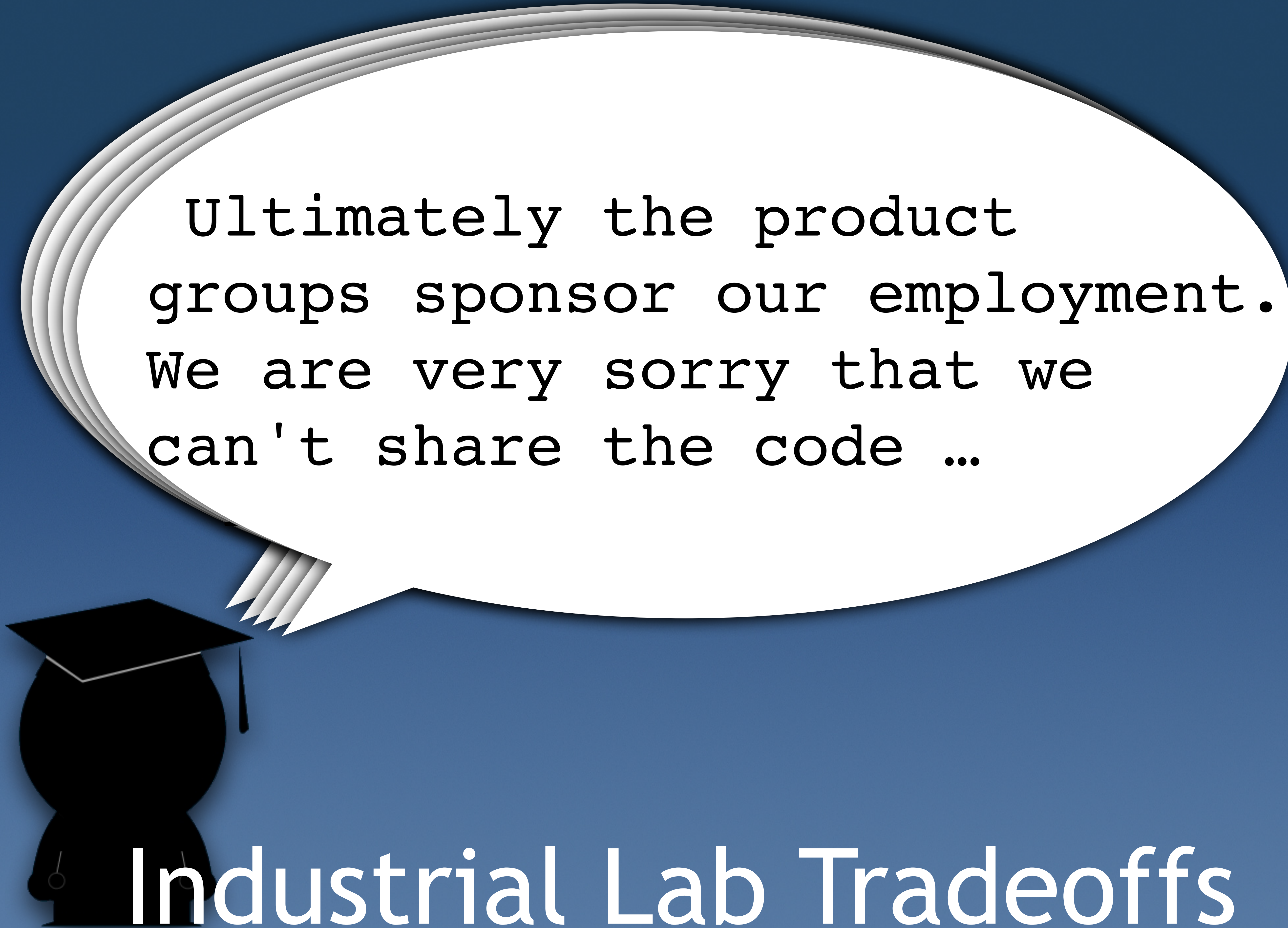
[Our] prototype … included many moving pieces that only student knew how to operate … **he left**.

Personnel Issues

… the server in which my implementation was stored had a **disk crash** … three disks crashed … Sorry for that.

Lost Code


[Therefore] we will not provide the source code outside the group.
Academic Tradeoffs



Industrial Lab Tradeoffs

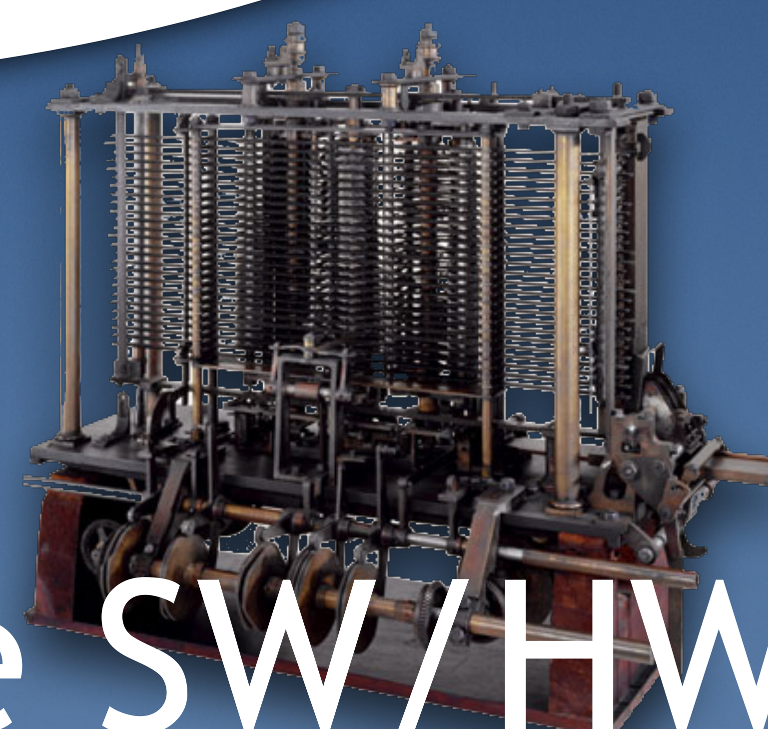
Unfortunately, the system sources are not meant to be open source (the code is partially property of three universities).

Proprietary Academic

and we also have a **software license agreement** that the University would need to sign.

Licensing Restrictions

… few people would manage to get it to work on new hardware.



Obsolete SW/HW

… based on earlier (bad) experience, we [want] to make sure that our implementation is not **used in situations that it was not meant for**.

Controlled Usage

… we have an agreement with the utility company, and we cannot release the code because of the potential **privacy risks** …

Privacy/Security

Design Issues

The code … is complete, but **hardly usable by anyone** other than the authors … due to our decision to use Template Haskell …

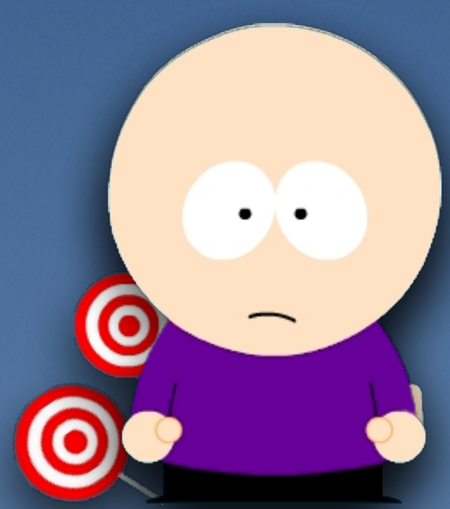
The Proposal

How Do We Fix This?

System

- Environment
- Benchmarks

Virtual Machine



- performance?
- security?
- longevity?

- if you build it, they still won’t come…

How Do We Fix This?

This measure was put in place to reassure authors who felt this would too radical a change to the process o evaluating conference paper

- 55 accepted papers
- 20 (36%) artifacts submitted for review
- 12 (22%) met or exceeded expectations

Dates

Paper decision notification: Feb 5, 2014
Artifacts due: Feb 10, 2014
Decisions announced: approx. Mar 15, 2014
Camera-ready due: Mar 20, 2014

How to Submit

Please read the guidelines on *what* to submit. Please upload your submission to EasyChair.

The Committee

The committee consists of several up-and-coming researchers with Eric Eide, Shriram Krishnamurthi, and Jan Vitek heading the process.

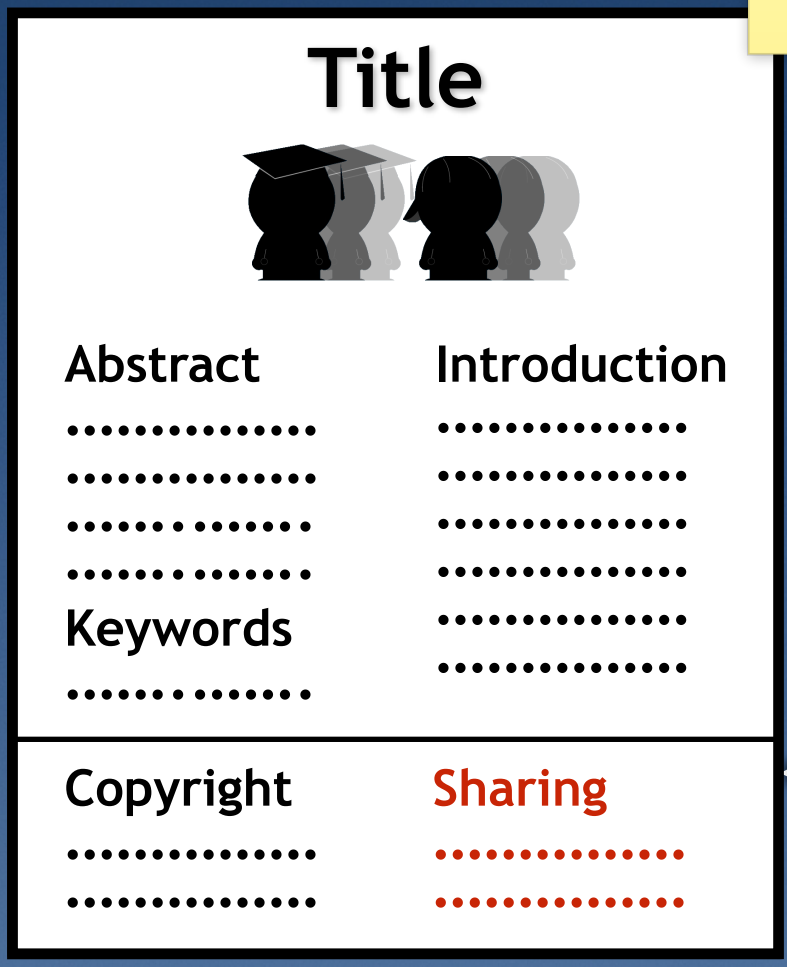
Process

Artifact evaluation is open *only* to accepted papers. This is intentional: it ensures that the AEC cannot influence whether or not a paper is accepted. This measure was put in place to reassure authors who felt this would be too radical a change to the process of evaluating conference paper submissions.

Of course, this doesn't mean you can't start getting ready! We have published the

A Modest Pro

[As all spheres of human activity are affected by the interplay between social structure and individual agency, sociology has gradually expanded its focus to further subjects, such as … , and the role of social activity in the development of scientific knowledge. http://en.wikipedia.org/wiki/Sociology].



Sharing

Low-cost, easily implementable, solution.

- artifact (code, data, media, docs, ...)
- availability (to download, access, install, run
- expense (free, fee, upgrade)
- distribution (source, binary, service)
- expiration date
- comment

sharing

```
http://reproducibility.cs.arizona.edu;
mailto:collberg@gmail.com;
code: access, free, source;
data: access, free, source, "sanitized";
support: L1, free, 2015-12-31;
```

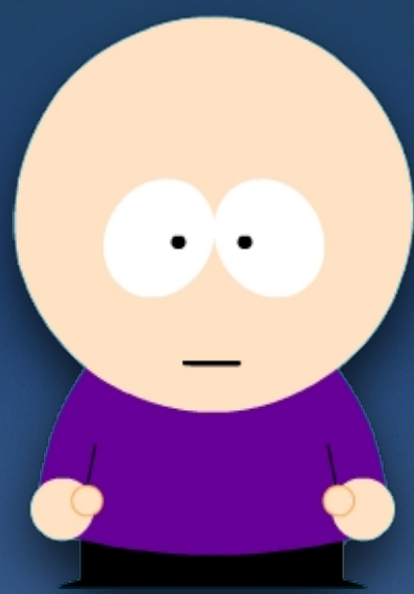
Location

Resource

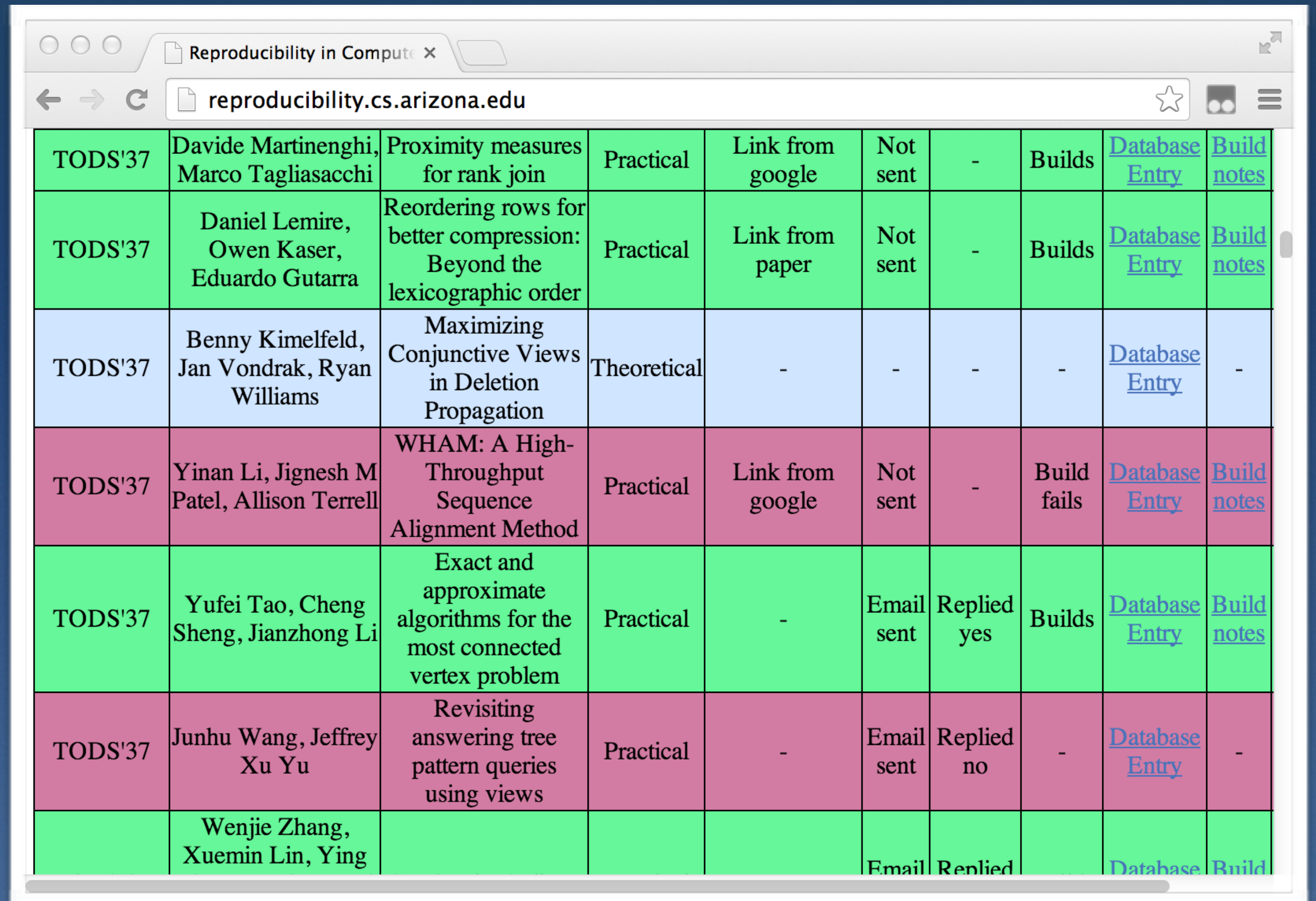
Support

Epilogue

What Happened Next?



Submitted Paper



TODS'37	Davide Martinenghi, Marco Tagliasacchi	Proximity measures for rank join	Practical	Link from google	Not sent	-	Builds	Database Entry	Build notes
TODS'37	Daniel Lemire, Owen Kaser, Eduardo Gutarra	Reordering rows for better compression: Beyond the lexicographic order	Practical	Link from paper	Not sent	-	Builds	Database Entry	Build notes
TODS'37	Benny Kimelfeld, Jan Vondrak, Ryan Williams	Maximizing Conjunctive Views in Deletion Propagation	Theoretical	-	-	-	-	Database Entry	-
TODS'37	Yinan Li, Jignesh M Patel, Allison Terrell	WHAM: A High-Throughput Sequence Alignment Method	Practical	Link from google	Not sent	-	Build fails	Database Entry	Build notes
TODS'37	Yufei Tao, Cheng Sheng, Jianzhong Li	Exact and approximate algorithms for the most connected vertex problem	Practical	-	Email sent	Replied yes	Builds	Database Entry	Build notes
TODS'37	Junhu Wang, Jeffrey Xu Yu	Revisiting answering tree pattern queries using views	Practical	-	Email sent	Replied no	-	Database Entry	-
	Wenjie Zhang, Xuemin Lin, Ying				Email	Replied		Database	Build

Technical Report

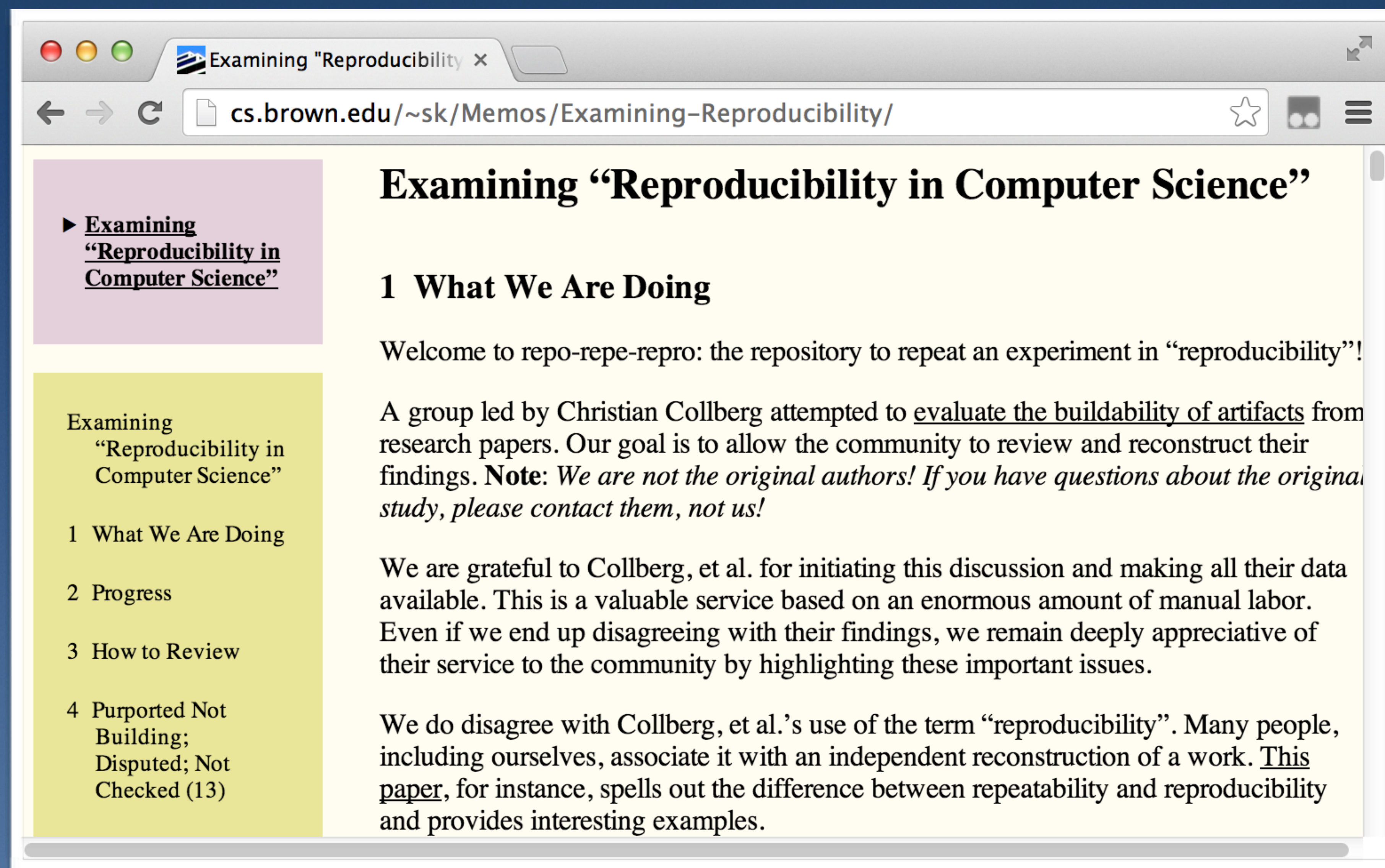
Dislike Us on Facebook!

Like you, I have been on the receiving end of stonewalling, silence, and maybe even lies when trying to get hold of code from others. Your emails … completely resonated with me.

et's
rter!
5!

* IRB - Institutional Review Board

Turnabout is Fair Play!



Examining "Reproducibility in Computer Science"

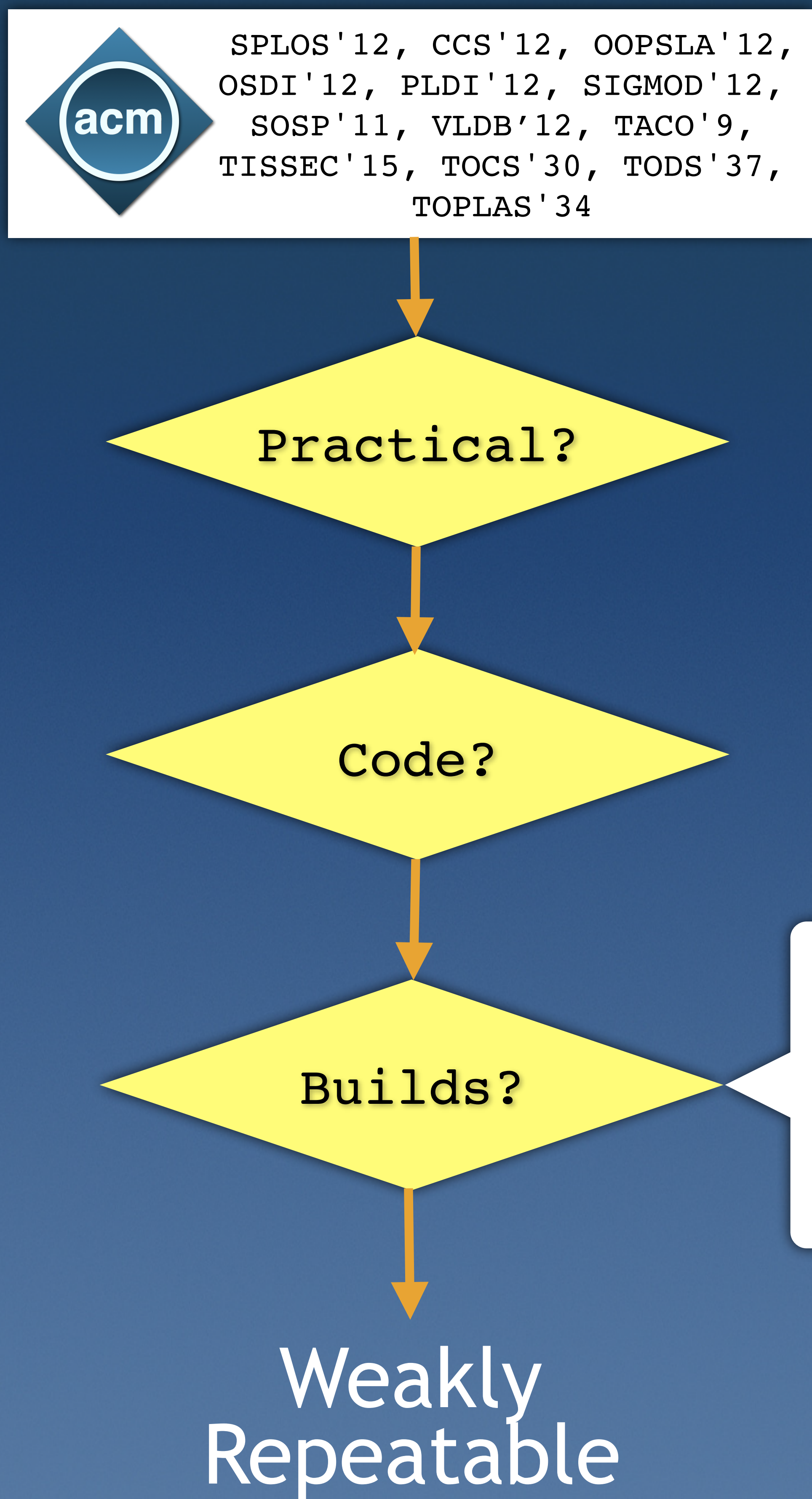
1 What We Are Doing

Welcome to repo-repe-repro: the repository to repeat an experiment in "reproducibility"!

A group led by Christian Collberg attempted to evaluate the buildability of artifacts from research papers. Our goal is to allow the community to review and reconstruct their findings. **Note**: *We are not the original authors! If you have questions about the original study, please contact them, not us!*

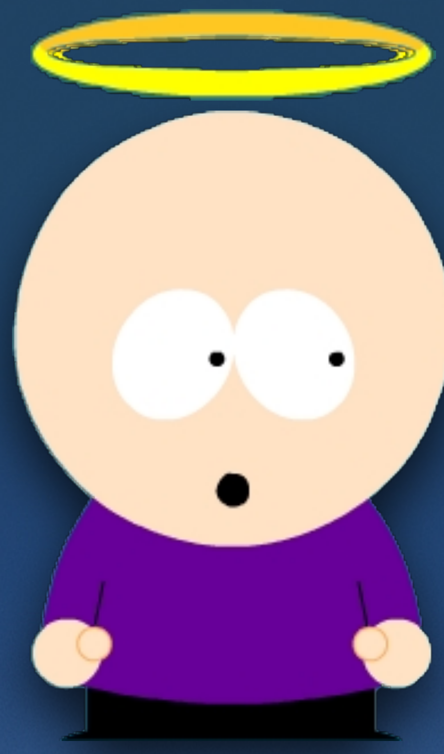
We are grateful to Collberg, et al. for initiating this discussion and making all their data available. This is a valuable service based on an enormous amount of manual labor. Even if we end up disagreeing with their findings, we remain deeply appreciative of their service to the community by highlighting these important issues.

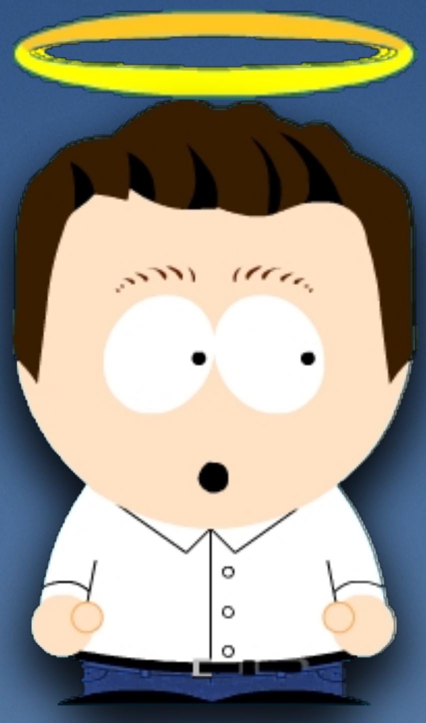
We do disagree with Collberg, et al.'s use of the term "reproducibility". Many people, including ourselves, associate it with an independent reconstruction of a work. This paper, for instance, spells out the difference between repeatability and reproducibility and provides interesting examples.



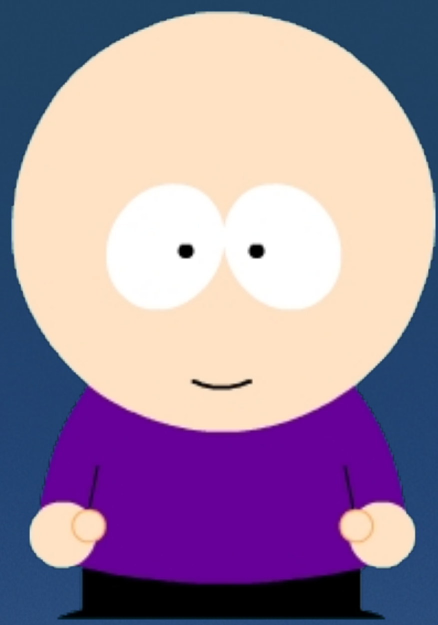
1. Reexamine failed builds
2. Request author feedback

Conclusions





1. Opening Gambit
2. Study
3. **Proposal**



1. Demanding everyone to share code always is unrealistic.

2. Sharing specifications are a low-cost alternative that can be implemented now.

3. We believe sharing specifications will be an incentive to authors to produce solid computational artifacts.


Questions?