Dare to Share: Risks and Rewards of Artifact Sharing in Computer Science

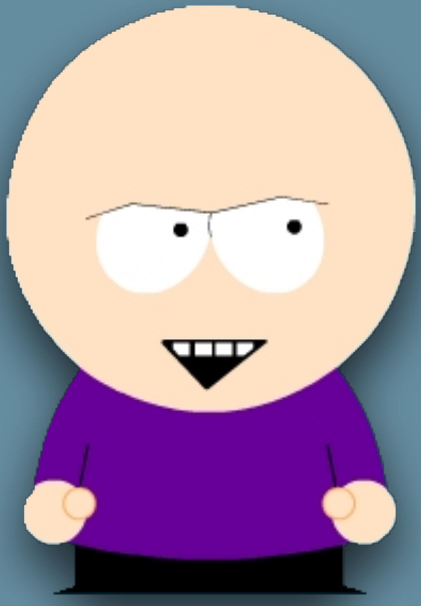
CPS-IoTBench 2019

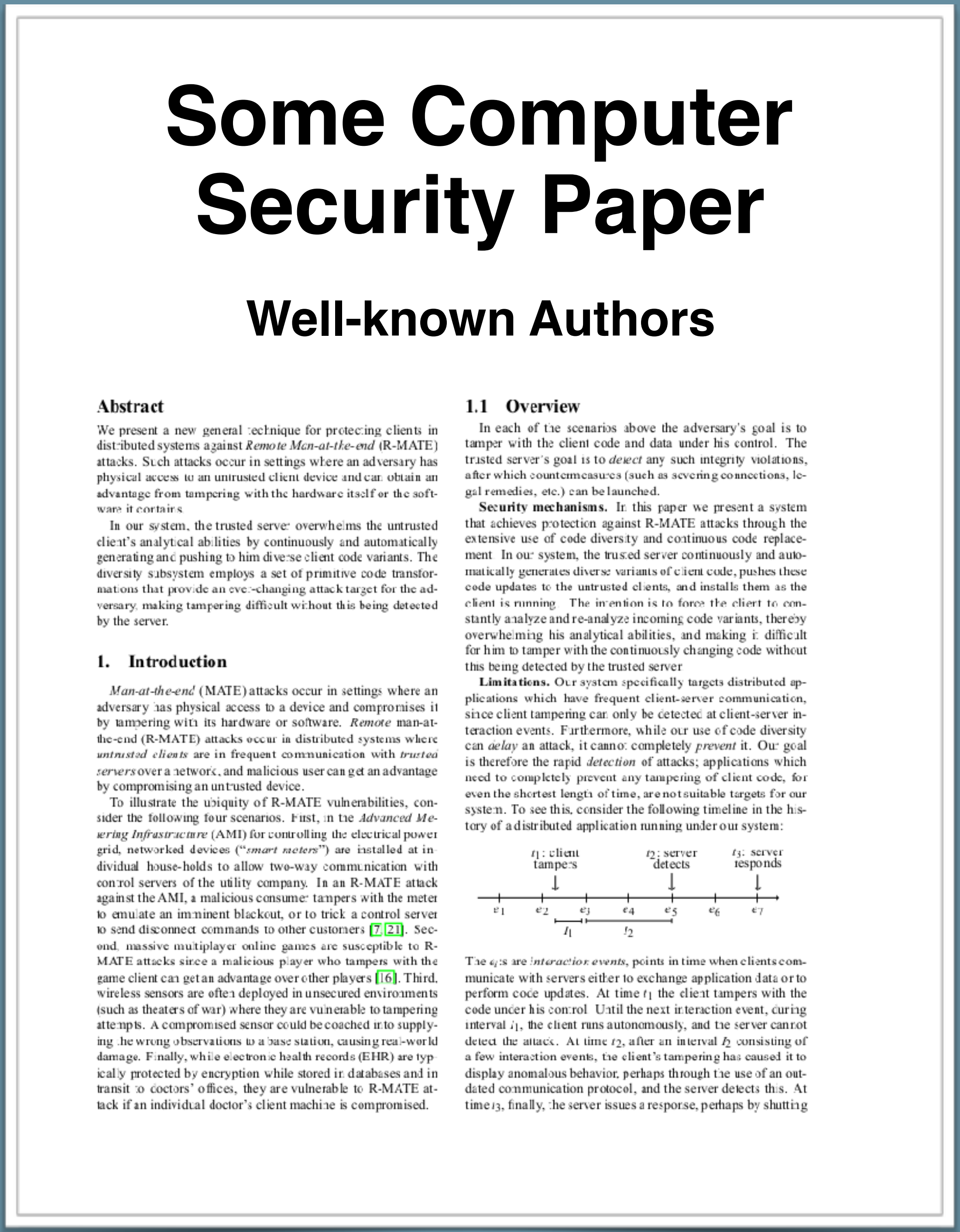
Christian Collberg Todd Proebsting
Keith Alcock
Department of Computer Science
University of Arizona

http://repeatability.cs.arizona.edu
http://findresearch.org

Supported by *the private foundation that must not be named*

1. Opening Gambit
2. The Deception Study
3. 8 Artifact Sharing Proposals
4. 8 Laws of Artifact Sharing
5. Risks and Rewards





Some Computer Security Paper

Well-known Authors

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.

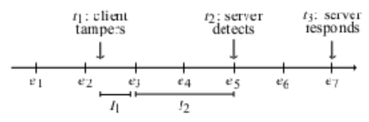
To illustrate the ubiquity of R-MATE vulnerabilities, consider the following four scenarios. First, in the *Advanced Metering Infrastructure* (AMI) for controlling the electrical power grid, networked devices ("*smart meters*") are installed at individual house-holds to allow two-way communication with 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

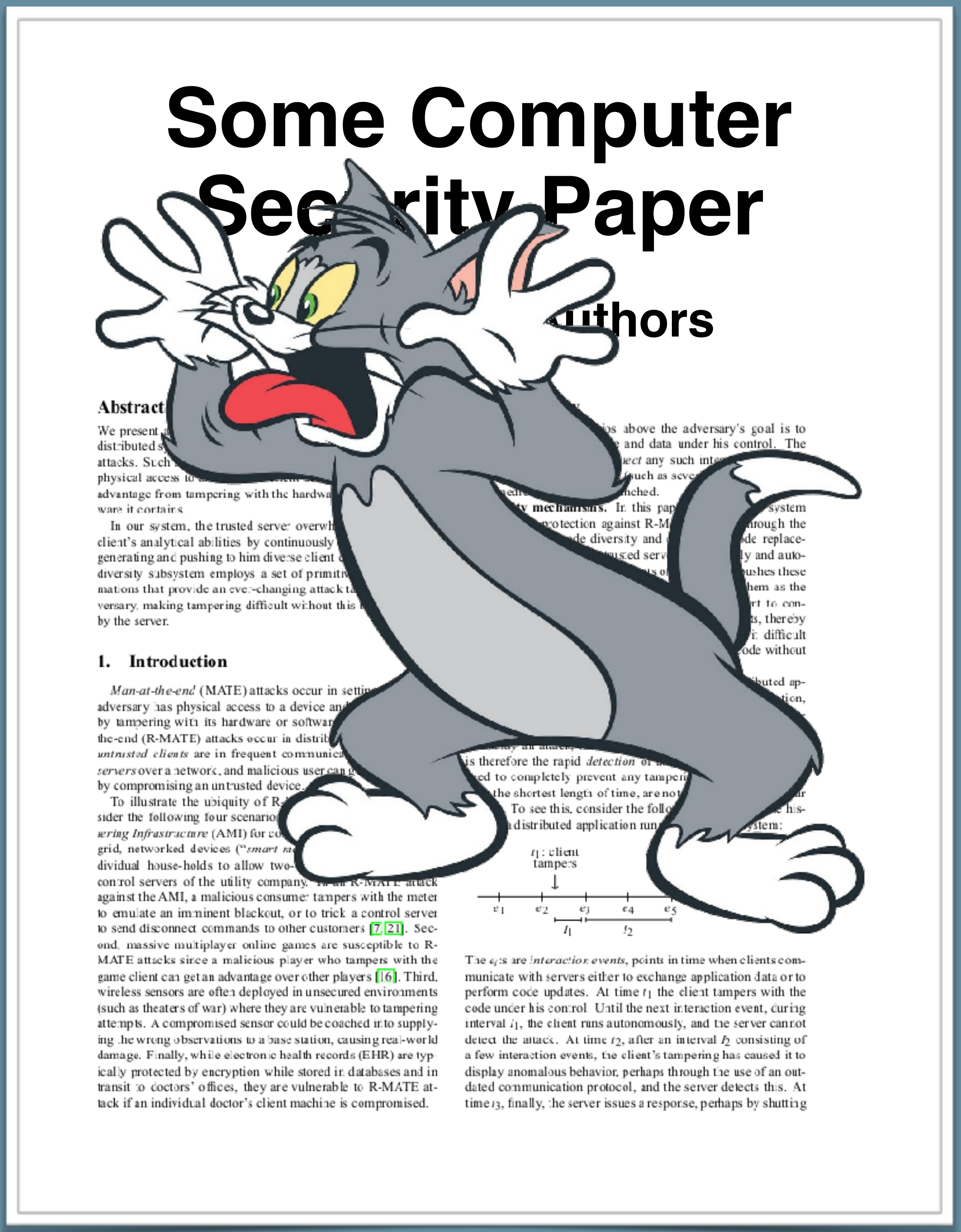
In each of the scenarios above the adversary's goal is to tamper with the client code and data under his control. The trusted server's goal is to *detect* any such integrity violations, after which countermeasures (such as severing connections, legal remedies, etc.) can be launched.

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



Some Computer
Sec rity Paper

uthors

Abstract

We present
distributed s
attacks. Such
physical access to
advantage from tampering with the hardwa
ware it contains.

In our system, the trusted server overwh
client's analytical abilities by continuously
generating and pushing to him diverse client
diversity subsystem employs a set of primiti
mations that provide an ever-changing attack ta
versary, making tampering difficult without this
by the server.

1. Introduction

Man-at-the-end (MATE) attacks occur in settin
adversary has physical access to a device an
by tampering with its hardware or softwar
the-end (R-MATE) attacks occur in distrib
untrusted clients are in frequent communica
servers over a network, and malicious user can g
by compromising an untrusted device.

To illustrate the ubiquity of R-
sider the following four scenario
ering Infrastructure (AMI) for co
grid, networked devices ("*smart m*
dividual house-holds to allow two-
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.

ios above the adversary's goal is to
and data under his control. The
ect any such inte
such as seve
nched.

ty mechanisms. In this pap system
protection against R-M hrough the
de diversity and de replace-
usted serv ly and auto-
ts of ushes these
hem as the
rt to con-
ts, thereby
it difficult
code without

ibuted ap-
tion,

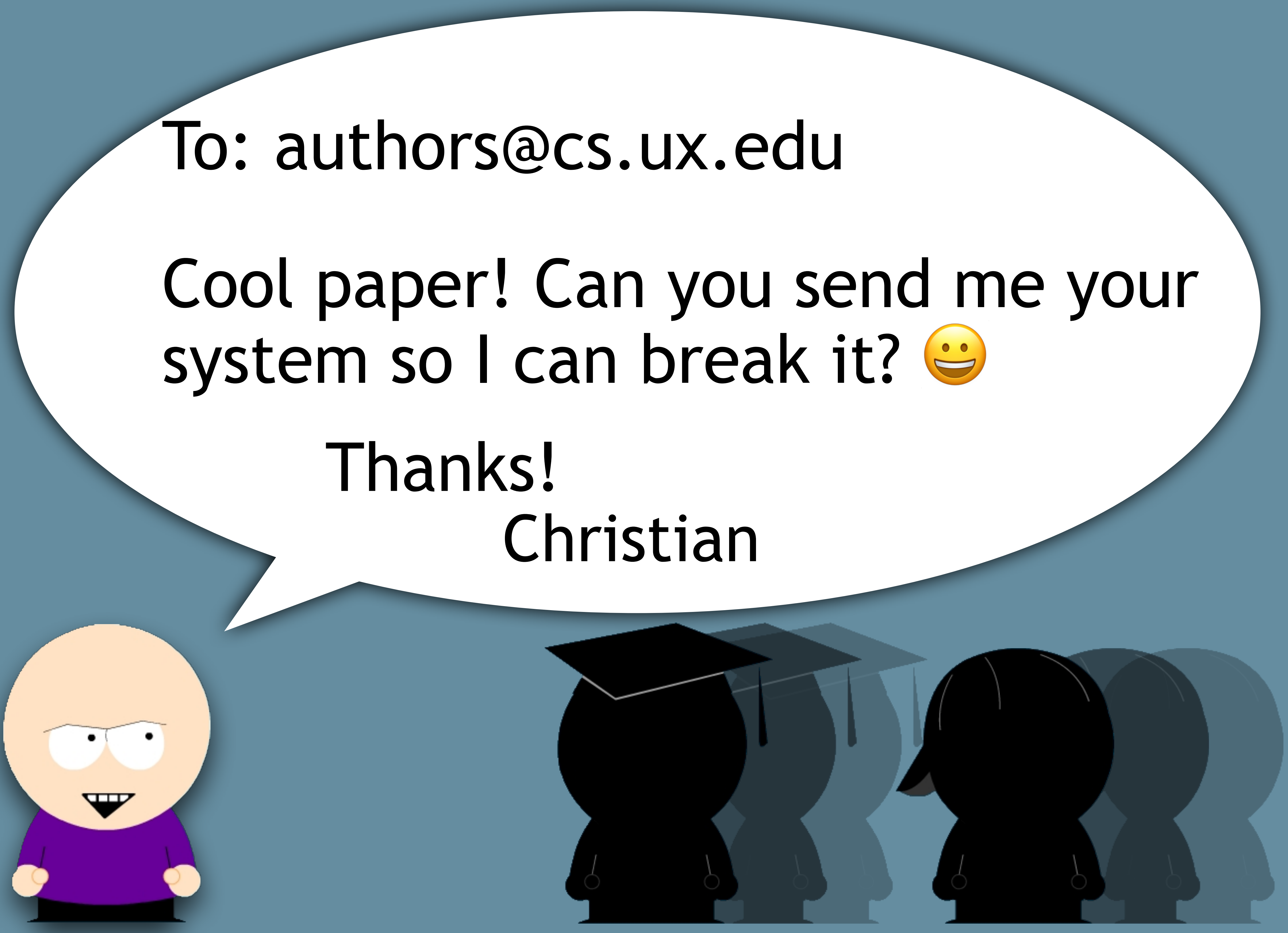
is therefore the rapid *detection* of
ed to completely prevent any tamperi
the shortest length of time, are not ir
To see this, consider the follo his-
a distributed application run ystem:

t_1: client
tampers

e_1 e_2 e_3 e_4 e_5

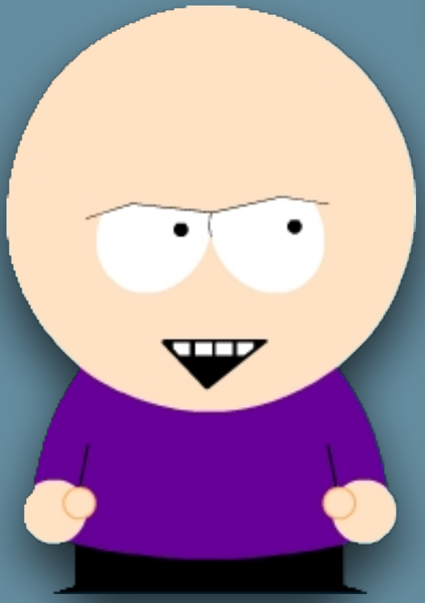
I_1 I_2

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

Cool paper! Can you send me your system so I can break it? 😀

Thanks!
Christian

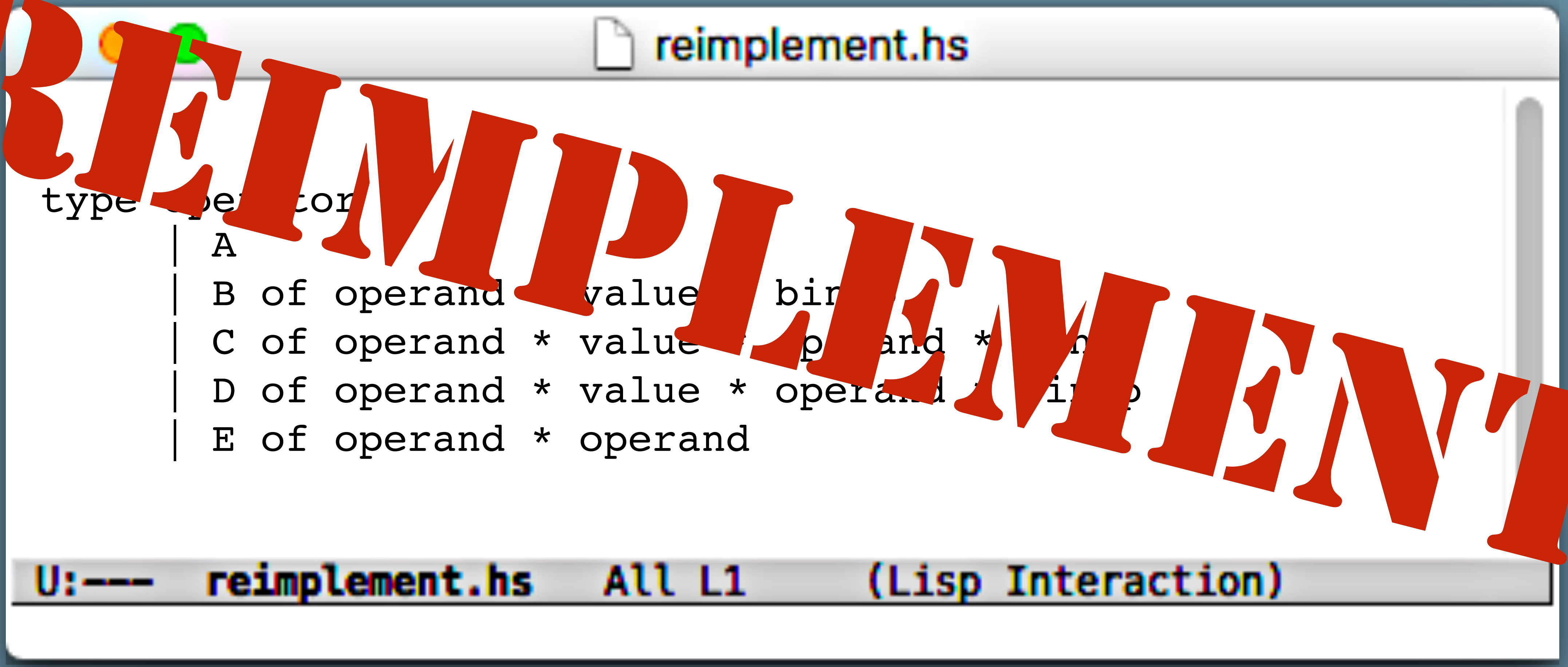


To: authors@cs.ux.edu

Cool paper! Can you send me your system so I can break it? 😀

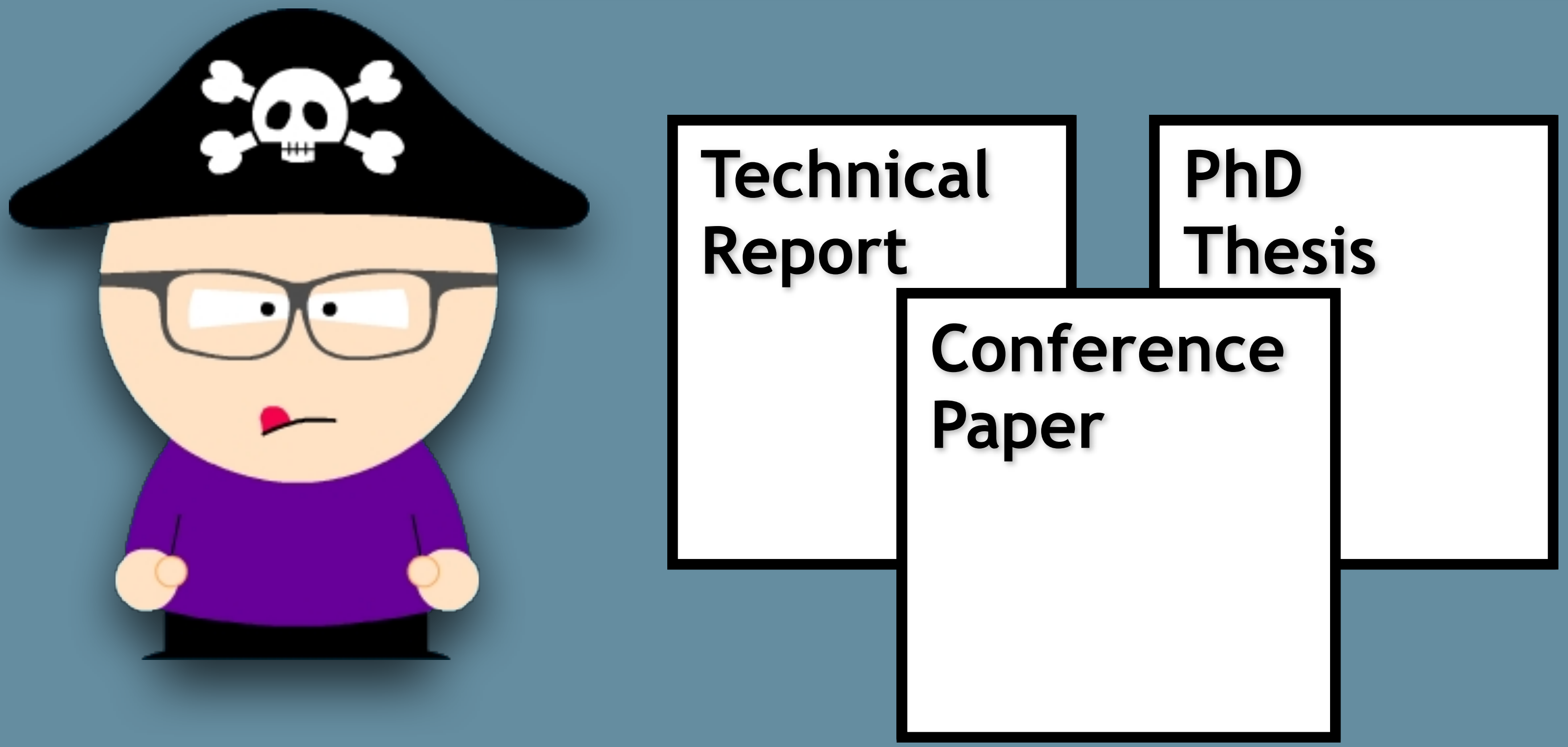
Thanks!
Christian

REIMPLEMENT!


reimplement.hs

```
type operator
      | A
      | B of operand * value * bin
      | C of operand * value * operand *
      | D of operand * value * operand *
      | E of operand * operand
```

```
U:---  reimplement.hs   All L1    (Lisp Interaction)
```



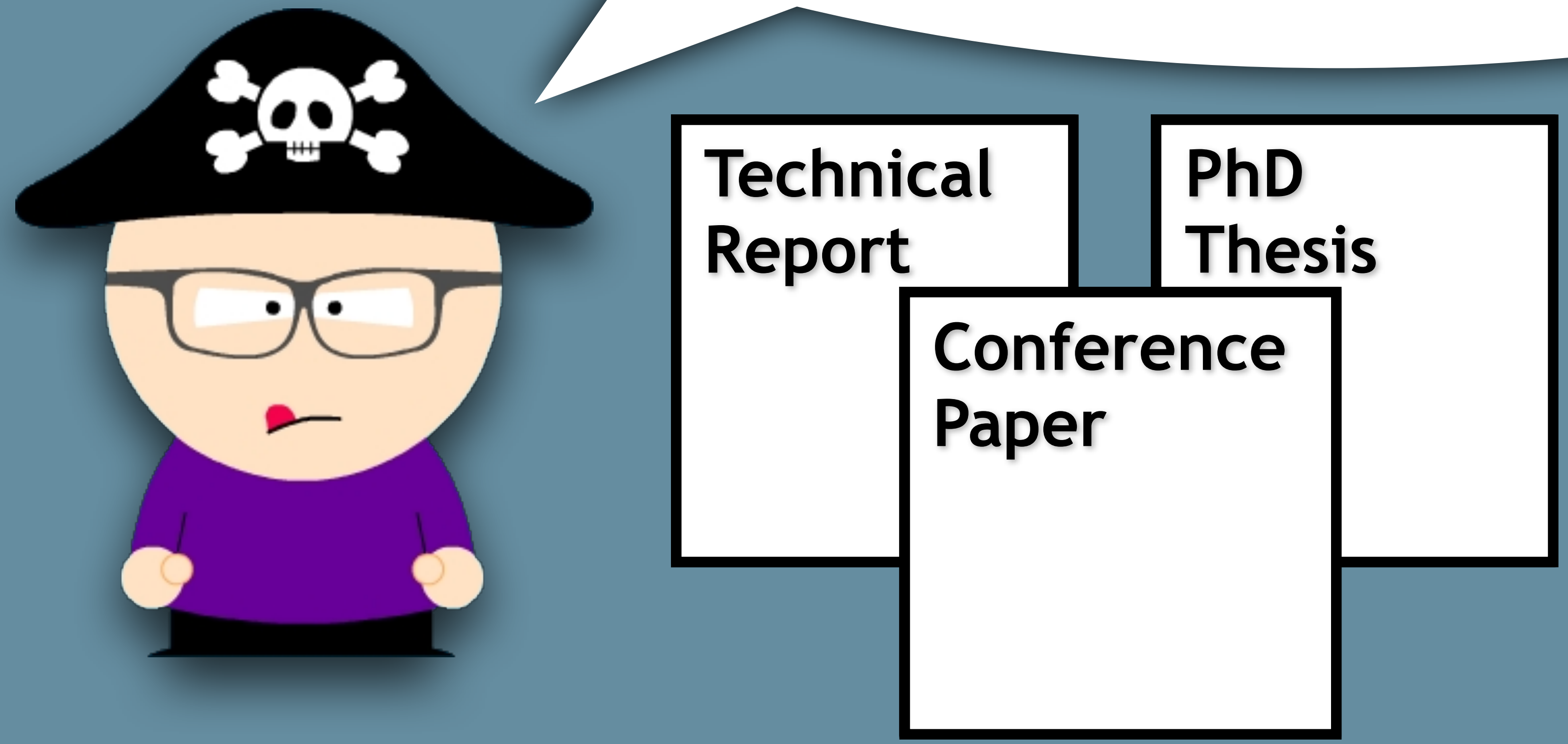
To: authors@cs.ux.edu

f: never used!

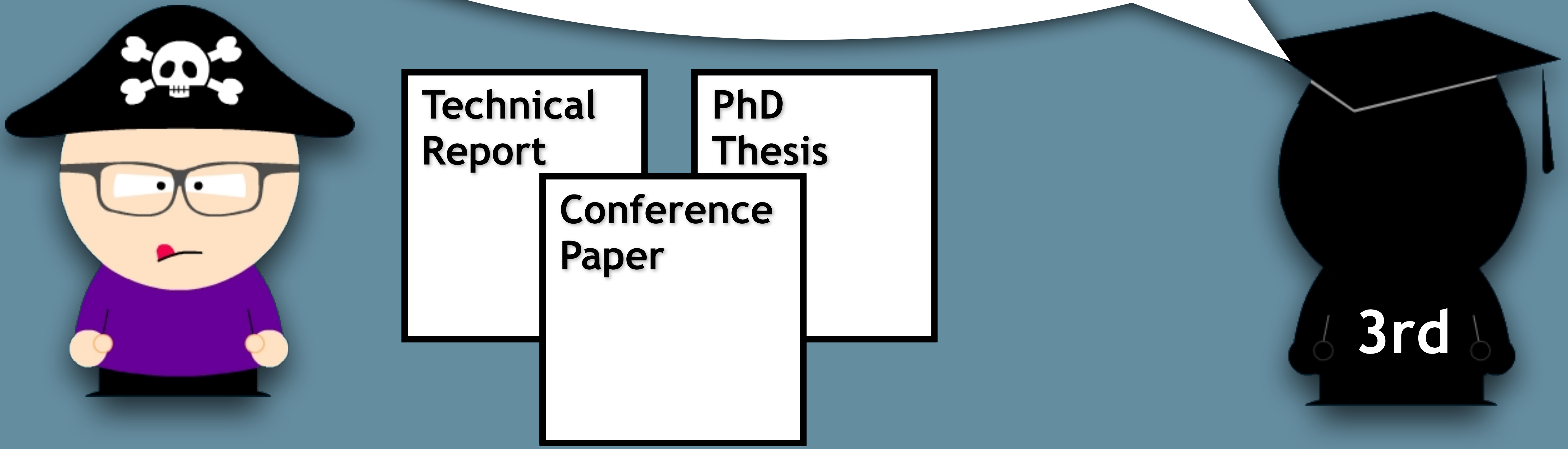
g: not defined!

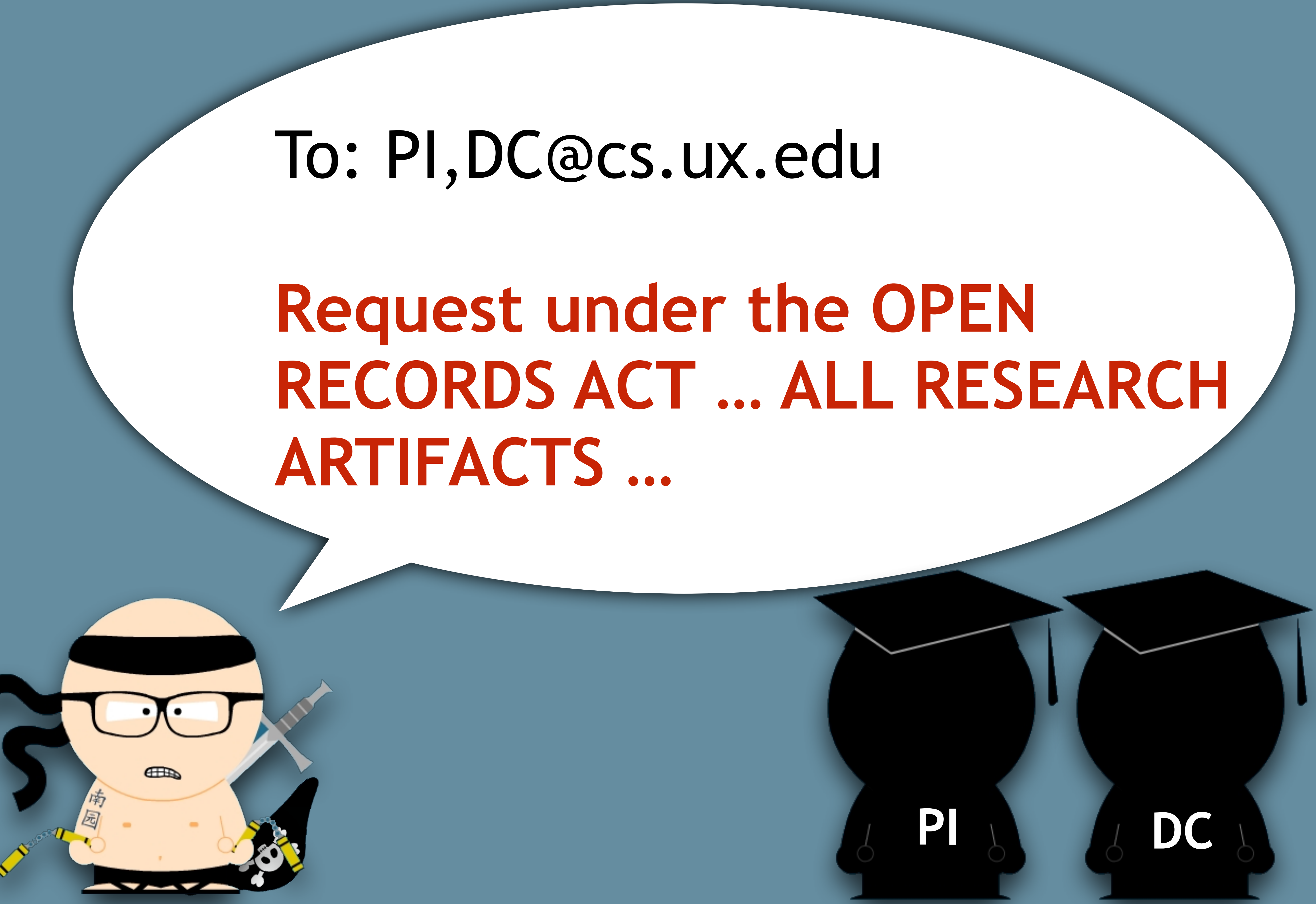
h: doesn’t type check!

i: different in TR and paper!



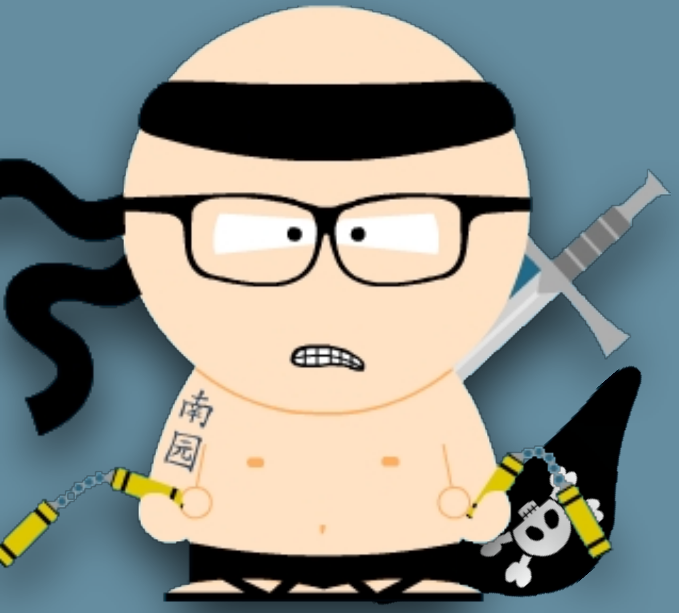
I … have **few recollections of the work**. [It was] like seeing a new paper for the first time.




To: PI,DC@cs.ux.edu
Request under the OPEN RECORDS ACT … ALL RESEARCH ARTIFACTS …
南
园
PI
DC

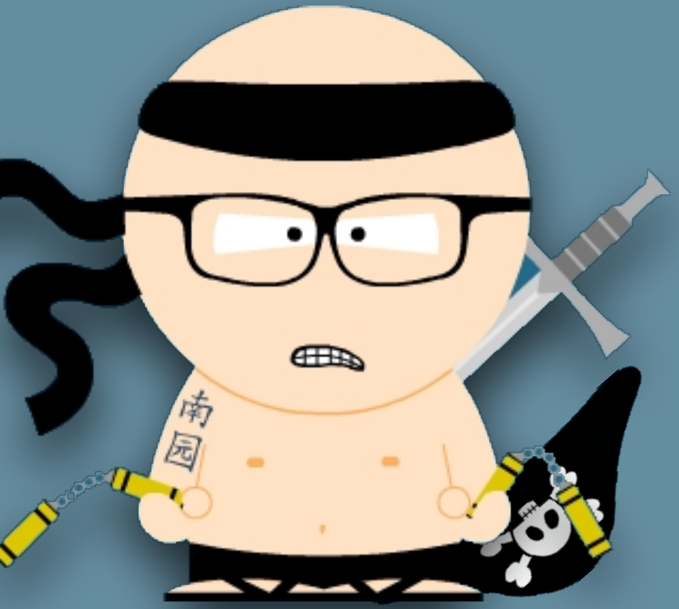
From: legal@cs.ux.edu

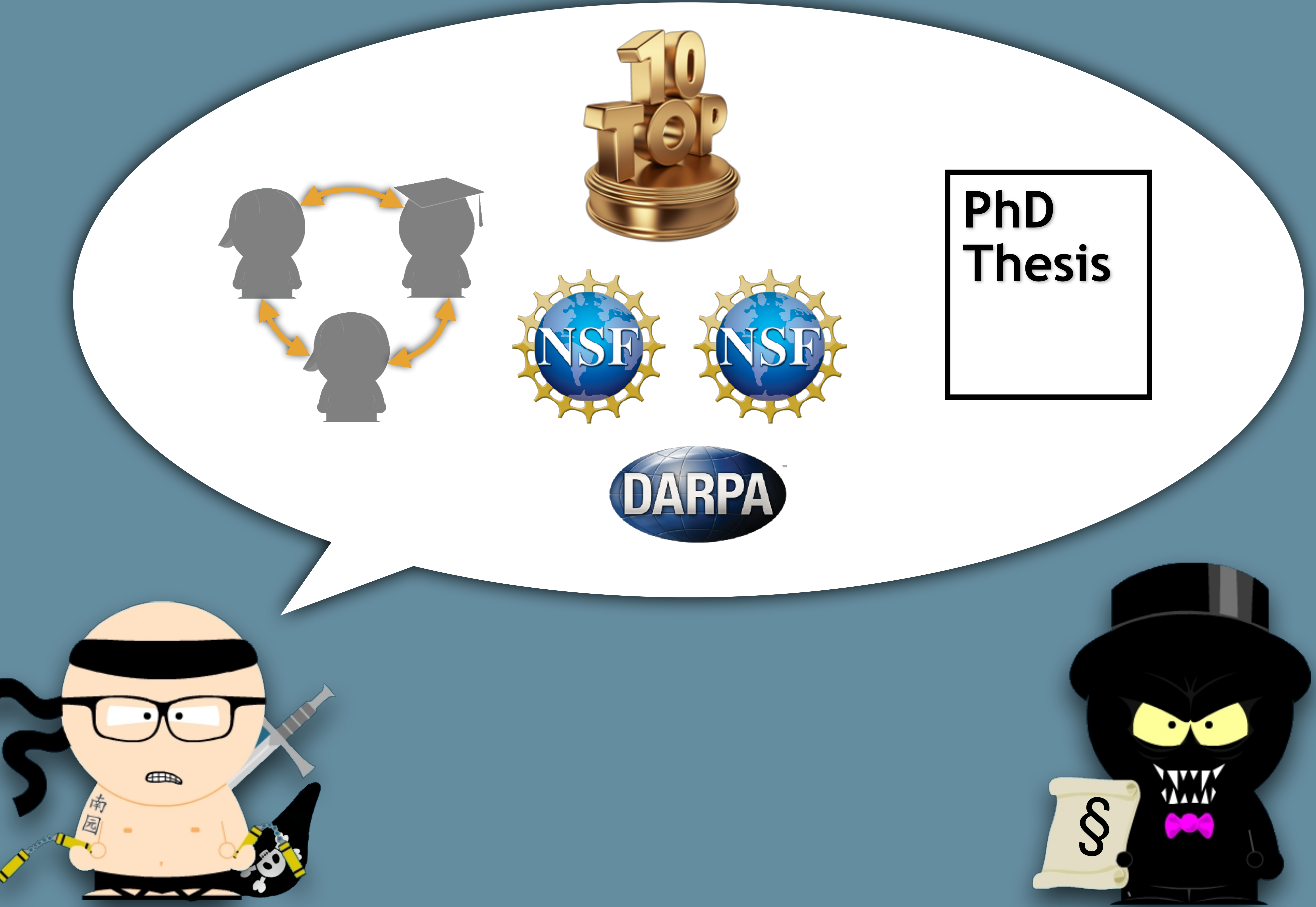
… to the extent such records may exist, **they will not be produced pursuant to ORA**.

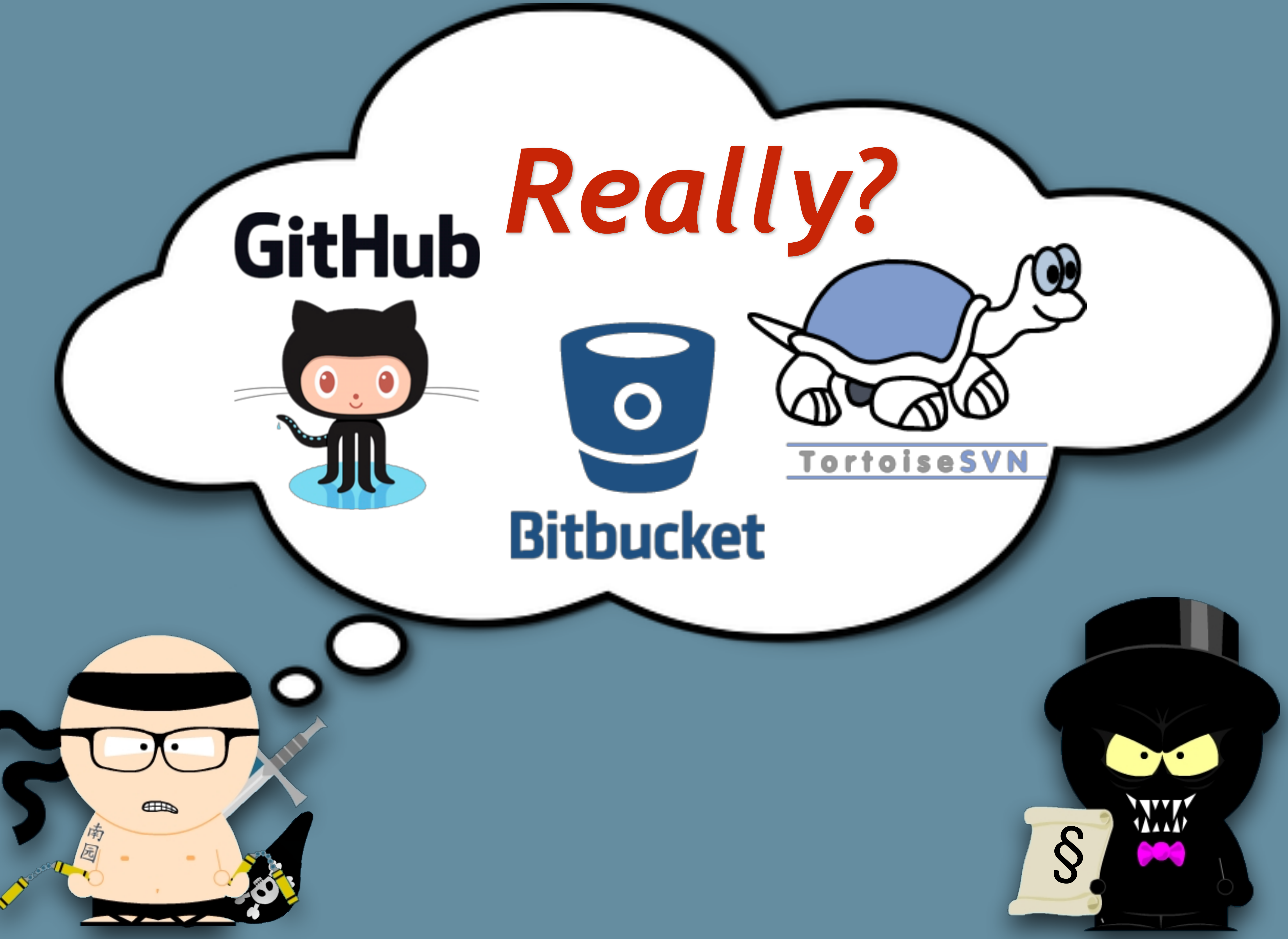


From: legal@cs.ux.edu

… and no, they don't exist…

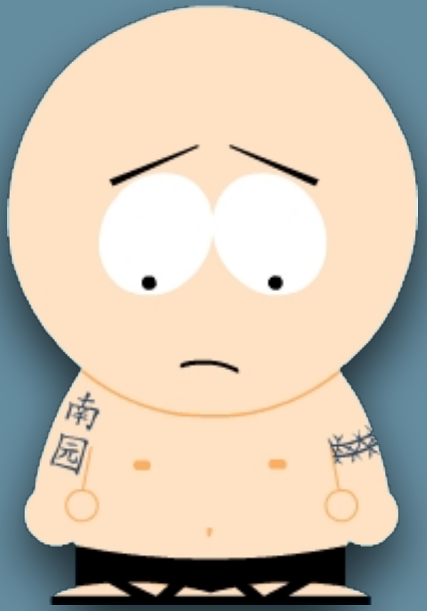



10
TOP
NSF
NSF
DARPA
PhD
Thesis
南园
§


Really?
GitHub
Bitbucket
TortoiseSVN
§

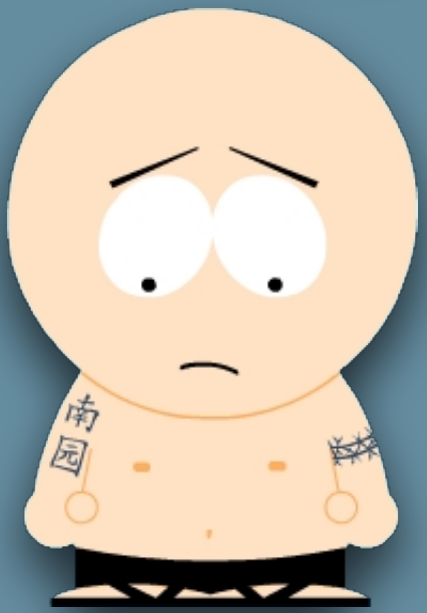
Pursuant to ORA, I request copies of **all electronic mail…**
§

… a total cost of **$2,263.66** to search for, retrieve, redact and produce such records.

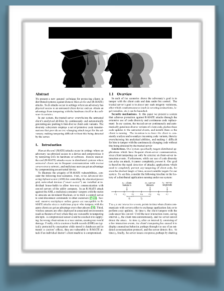


Grant application #: [redacted]

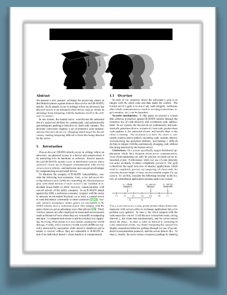
We will also make our data and software available to the research community **when appropriate**.



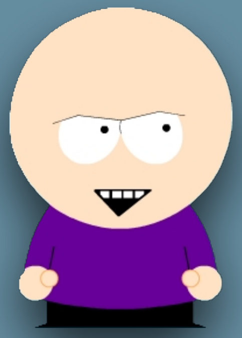
Read paper

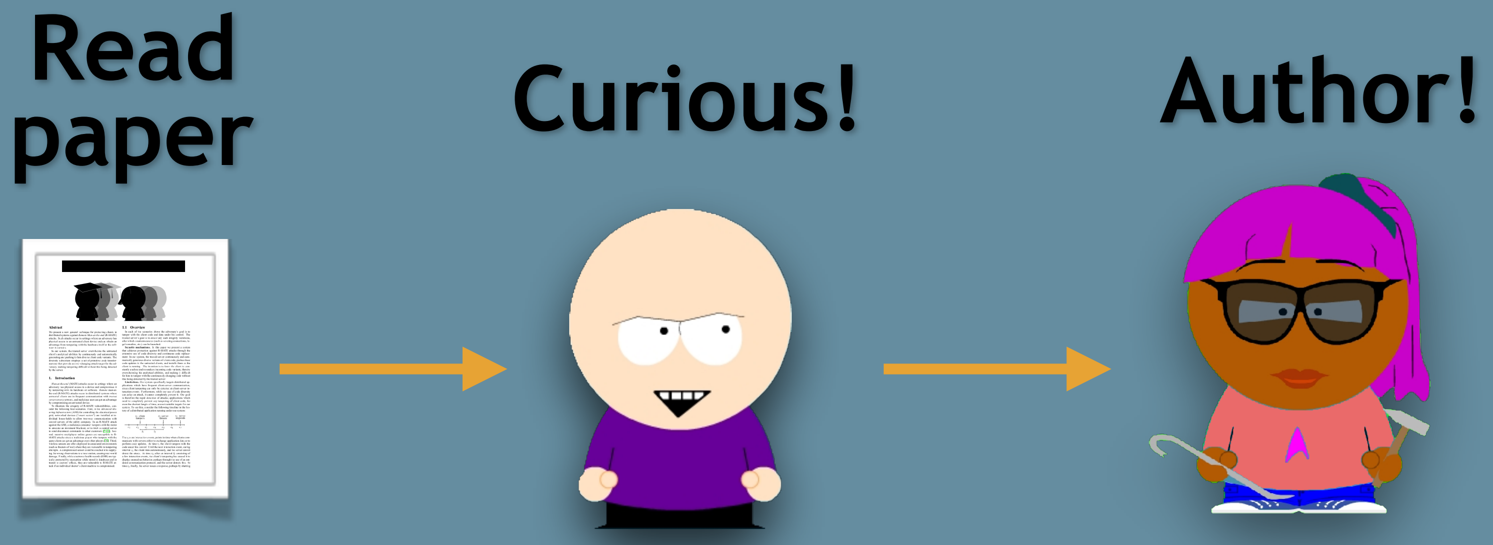


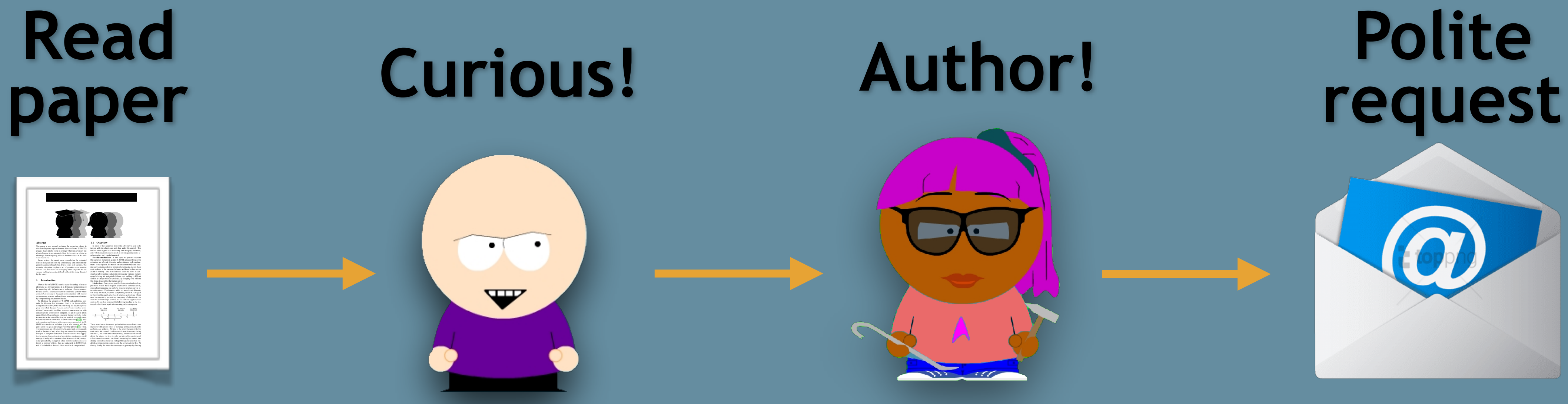
Read paper

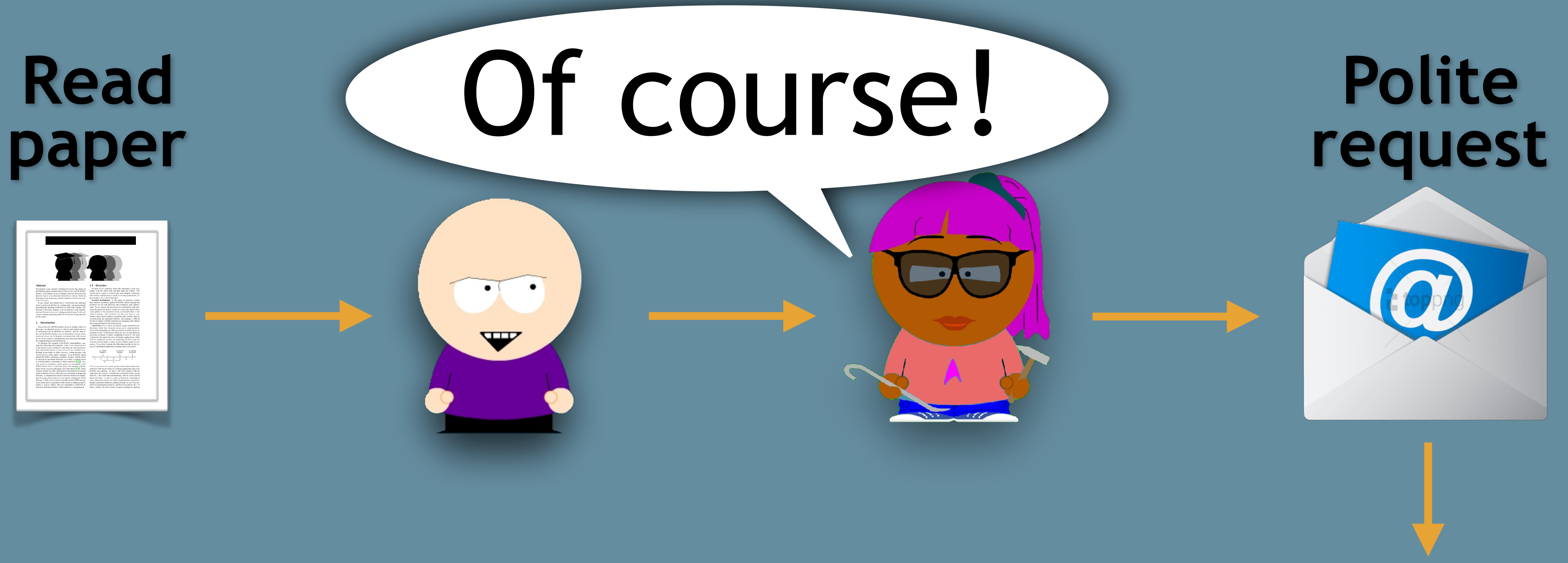


Curious!



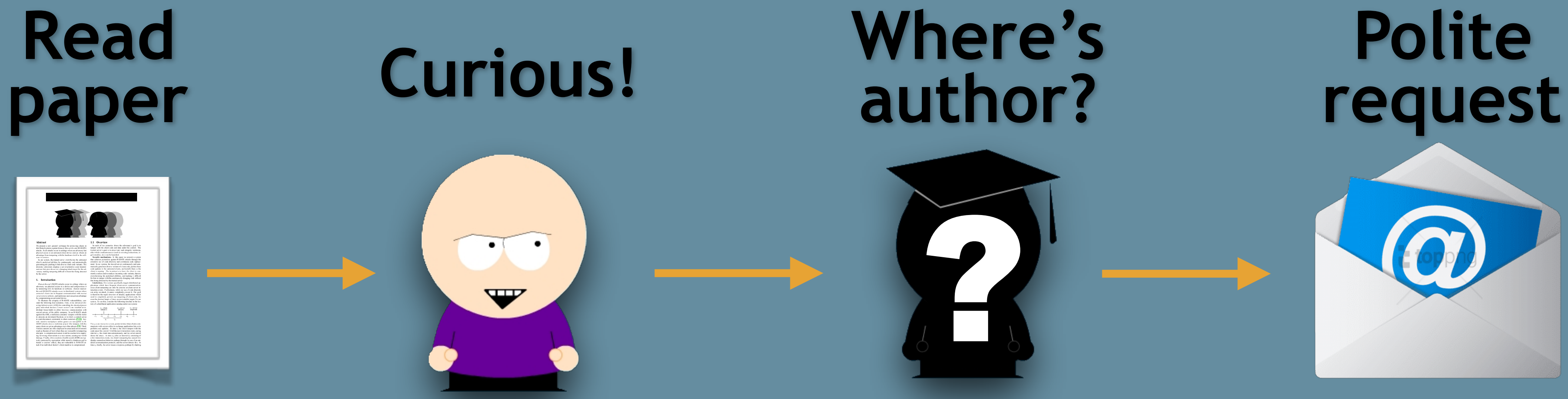

Read paper
Curious!
Author!

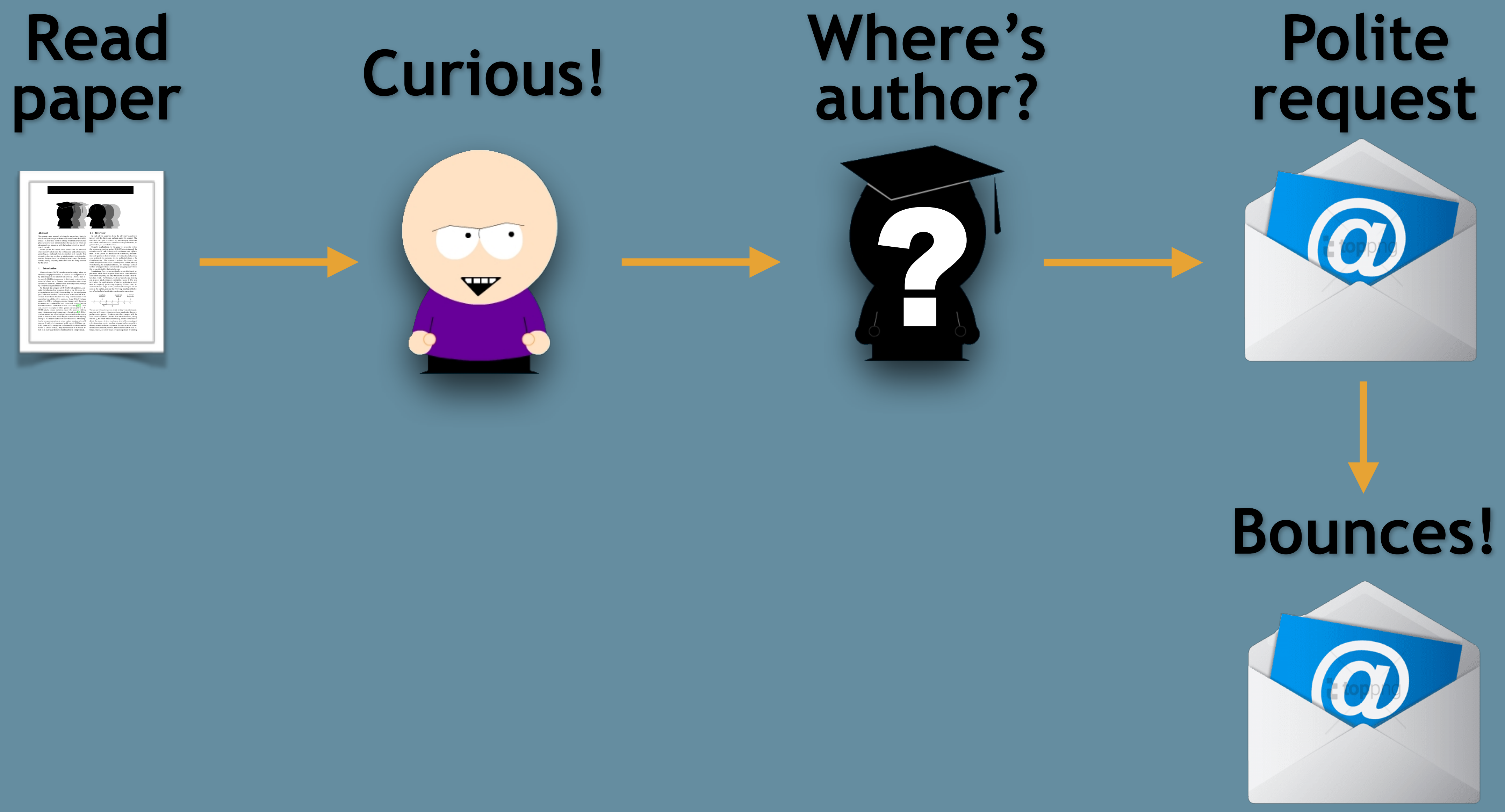

Read paper
Curious!
Author!
Polite request

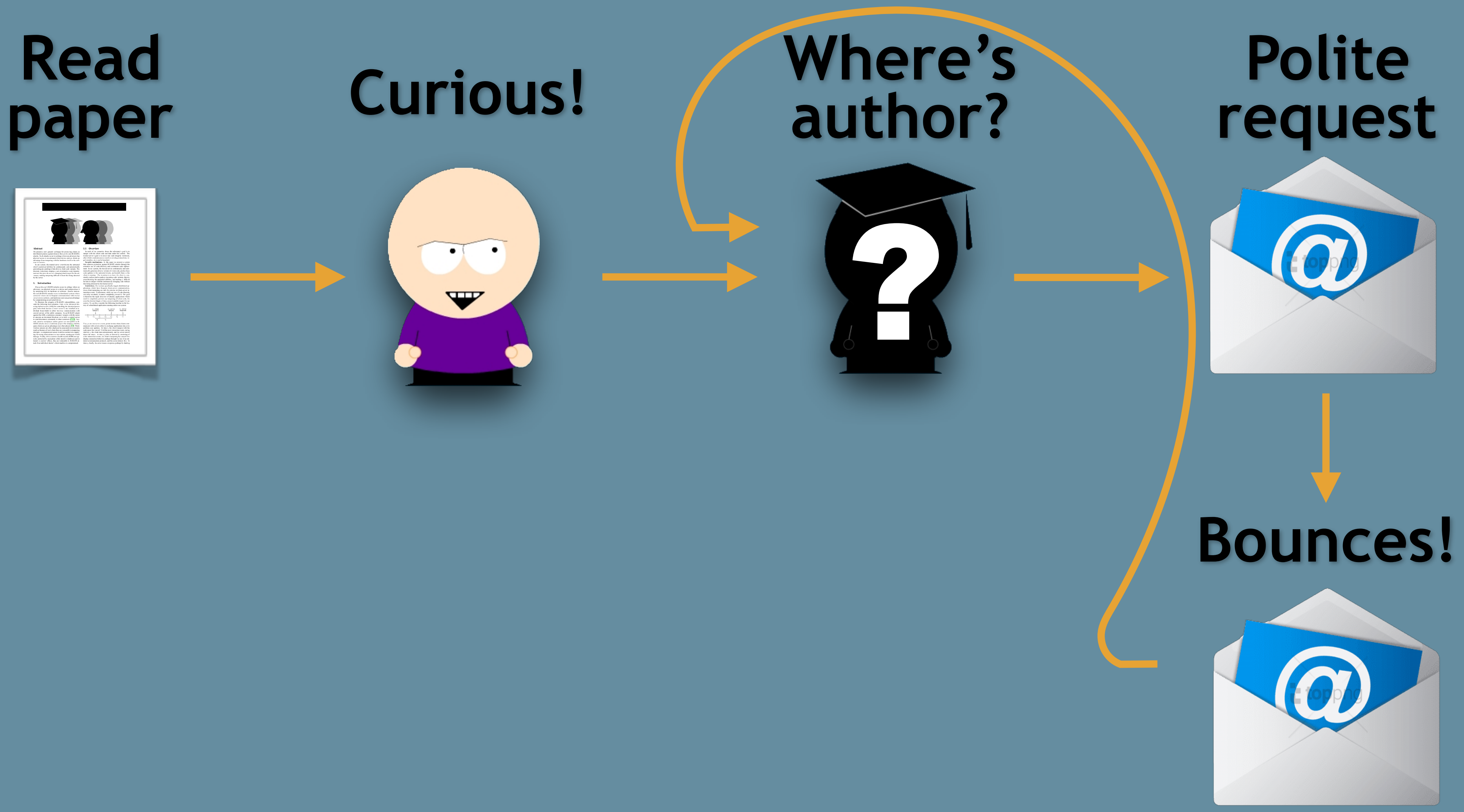


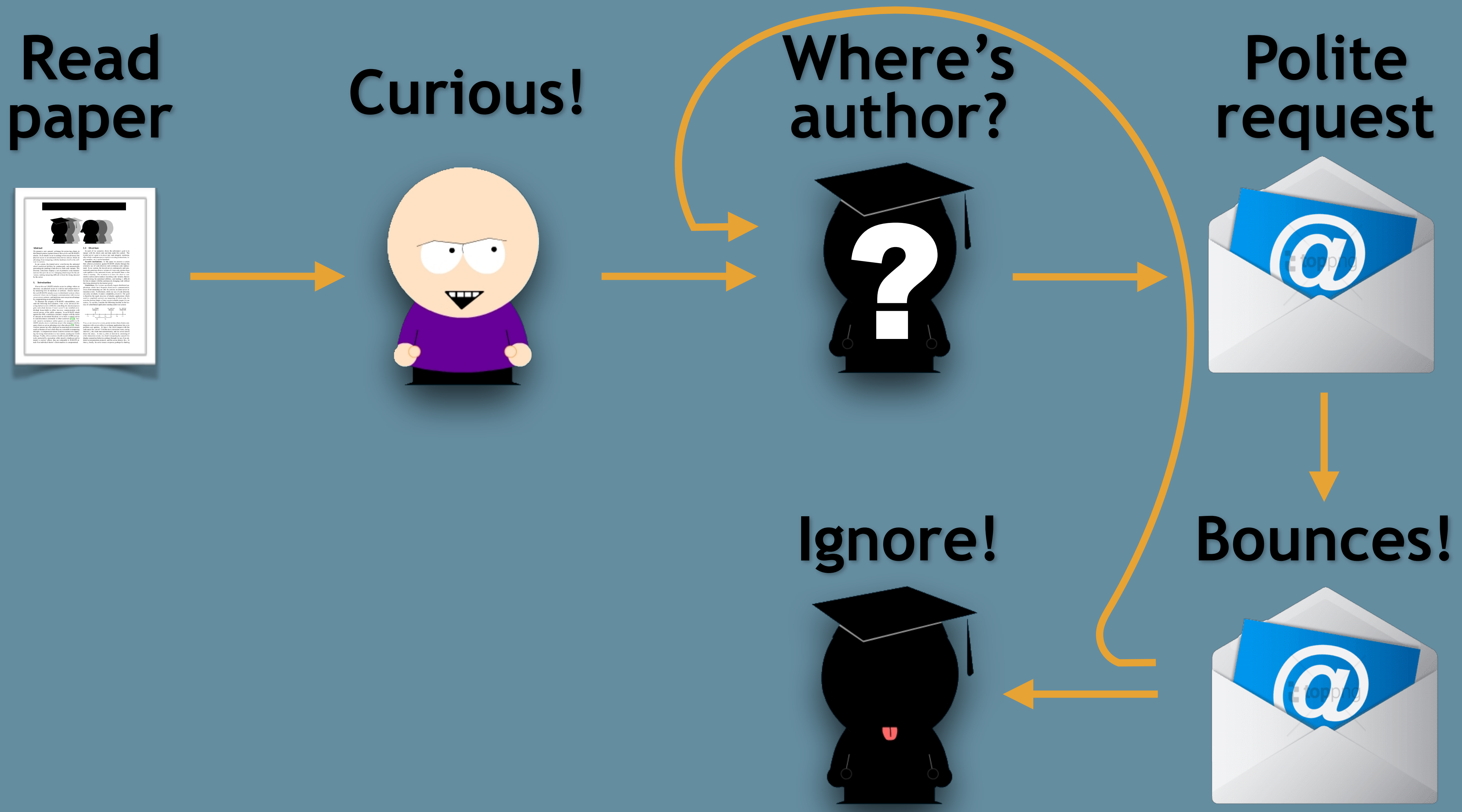
EVERYTHING_YOU_NEED.zip

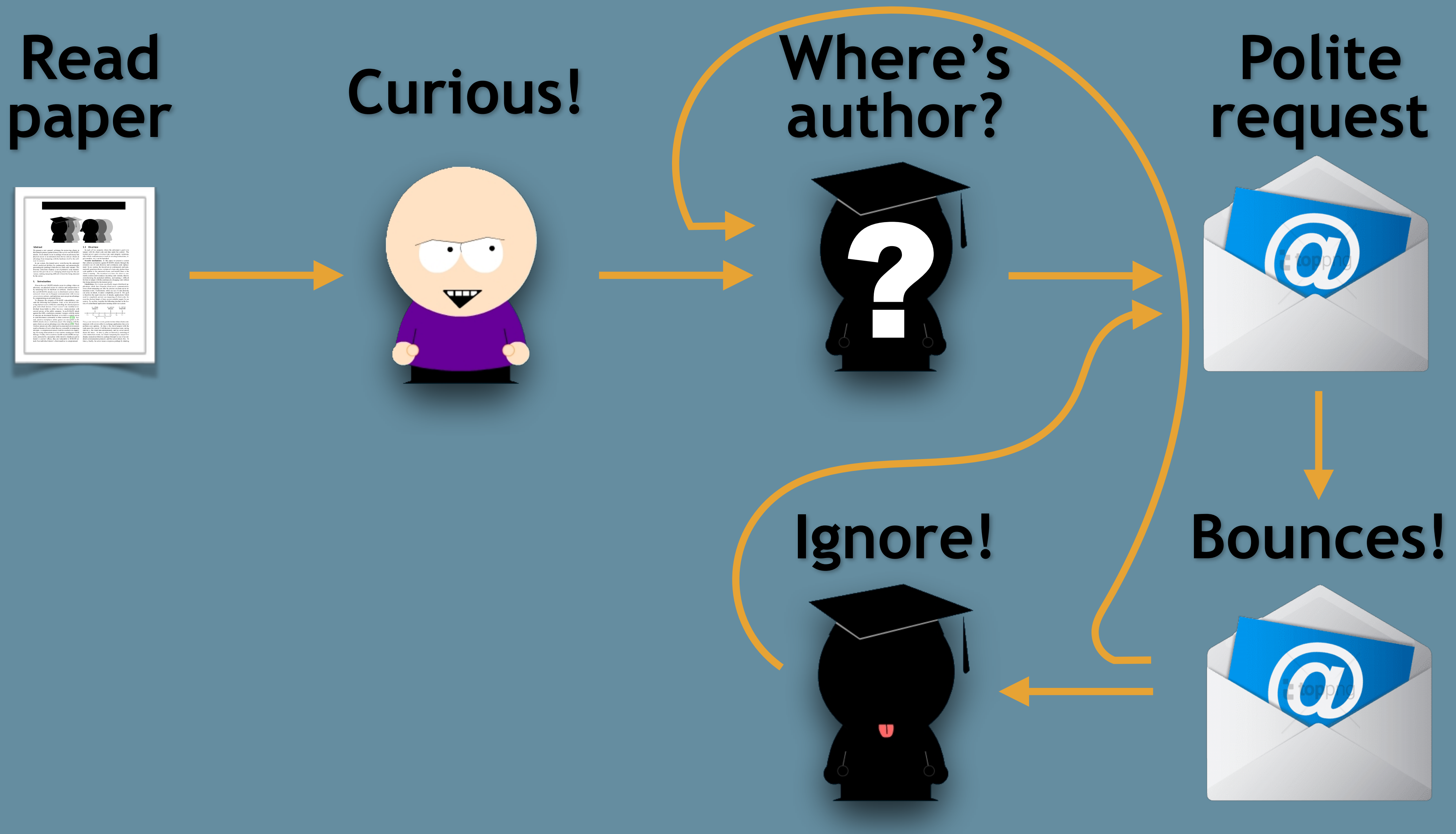
```
├── src
│   ├── Makefile
│   └── *.c
├── lib
│   ├── dependencies.txt
│   └── *.a
└── doc
    ├── README
    └── INSTALL
```

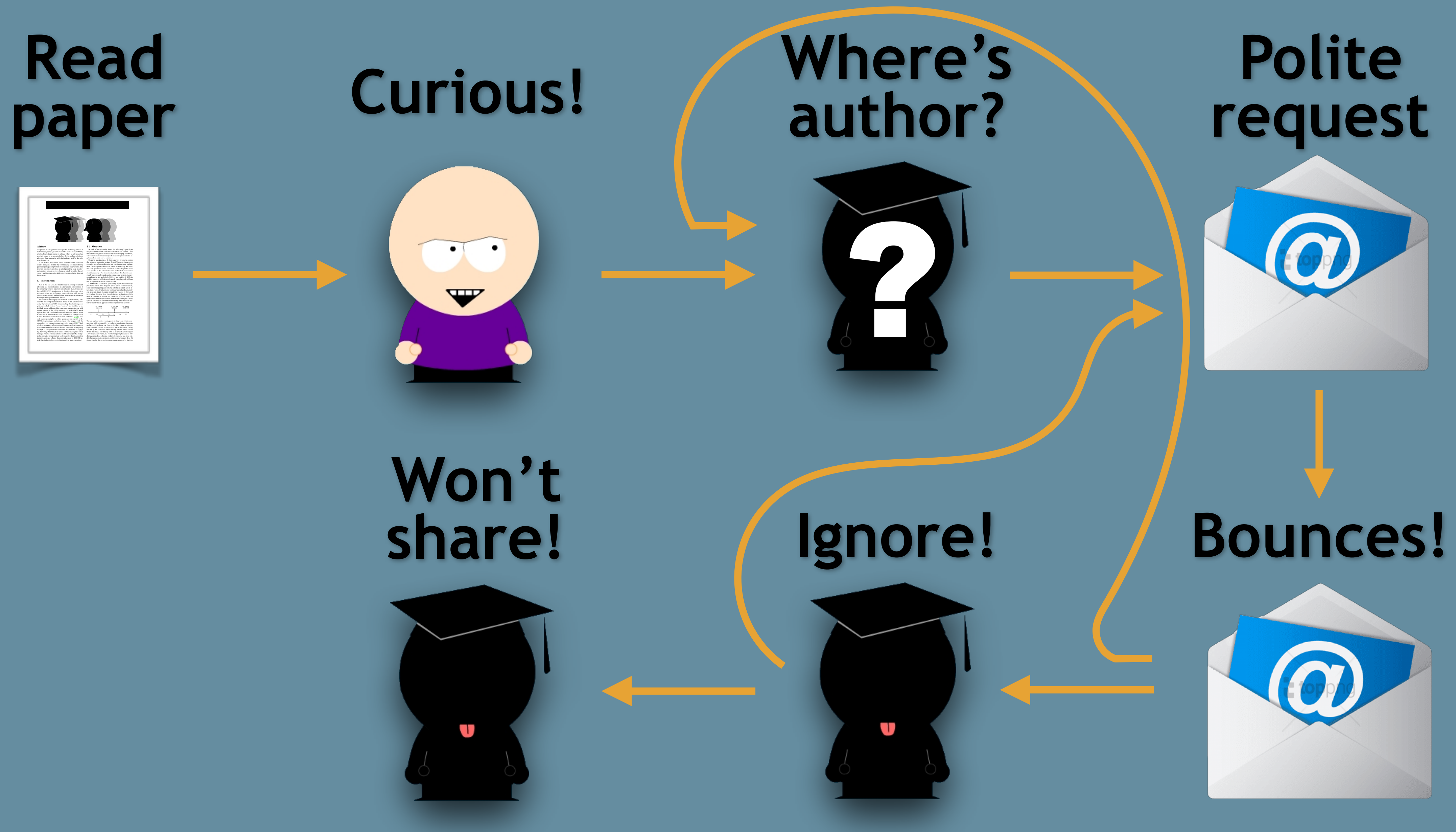

Read paper
Curious!
Where's author?
Polite request

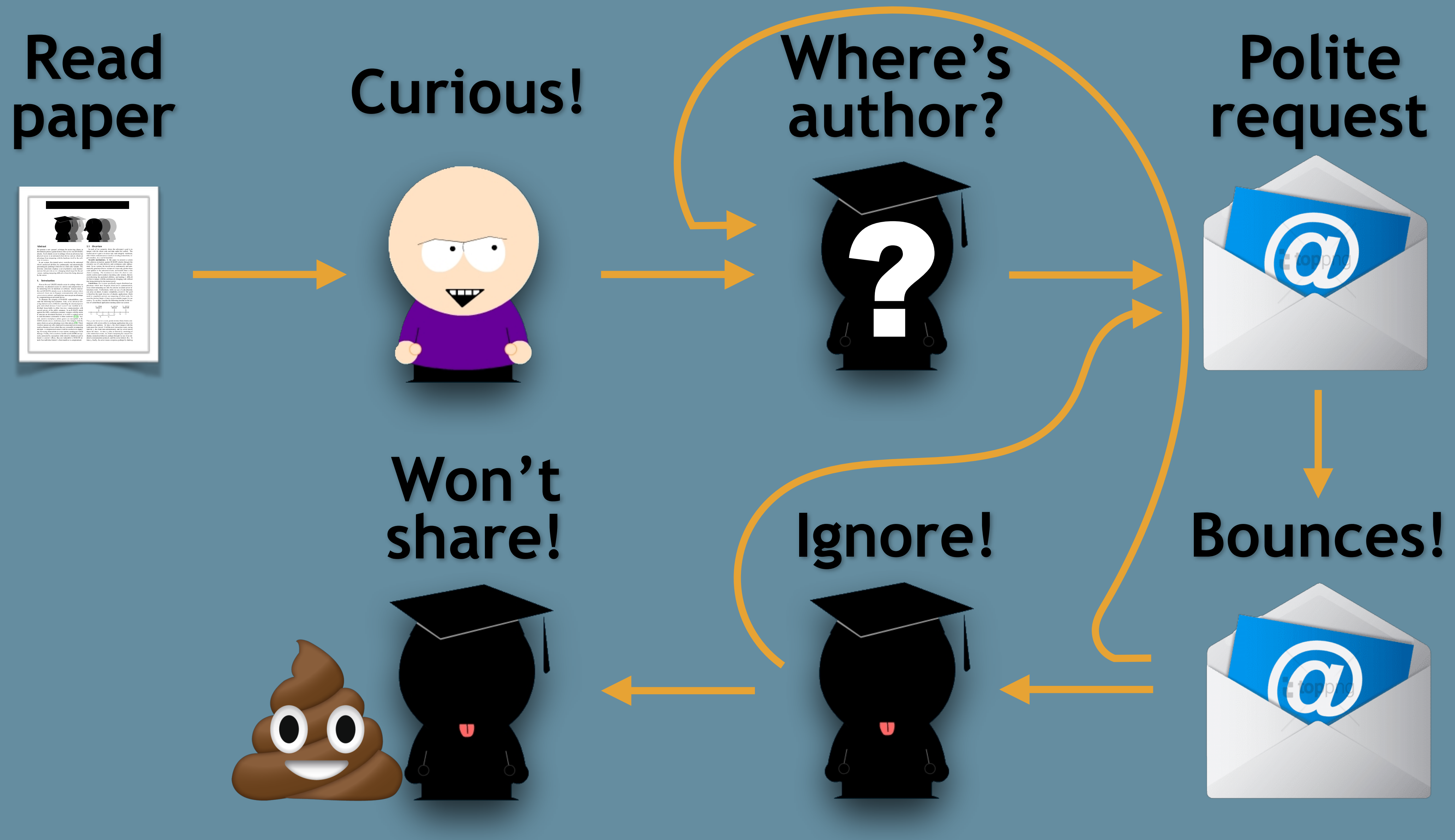

Read paper
Curious!
Where's author?
Polite request
Bounces!

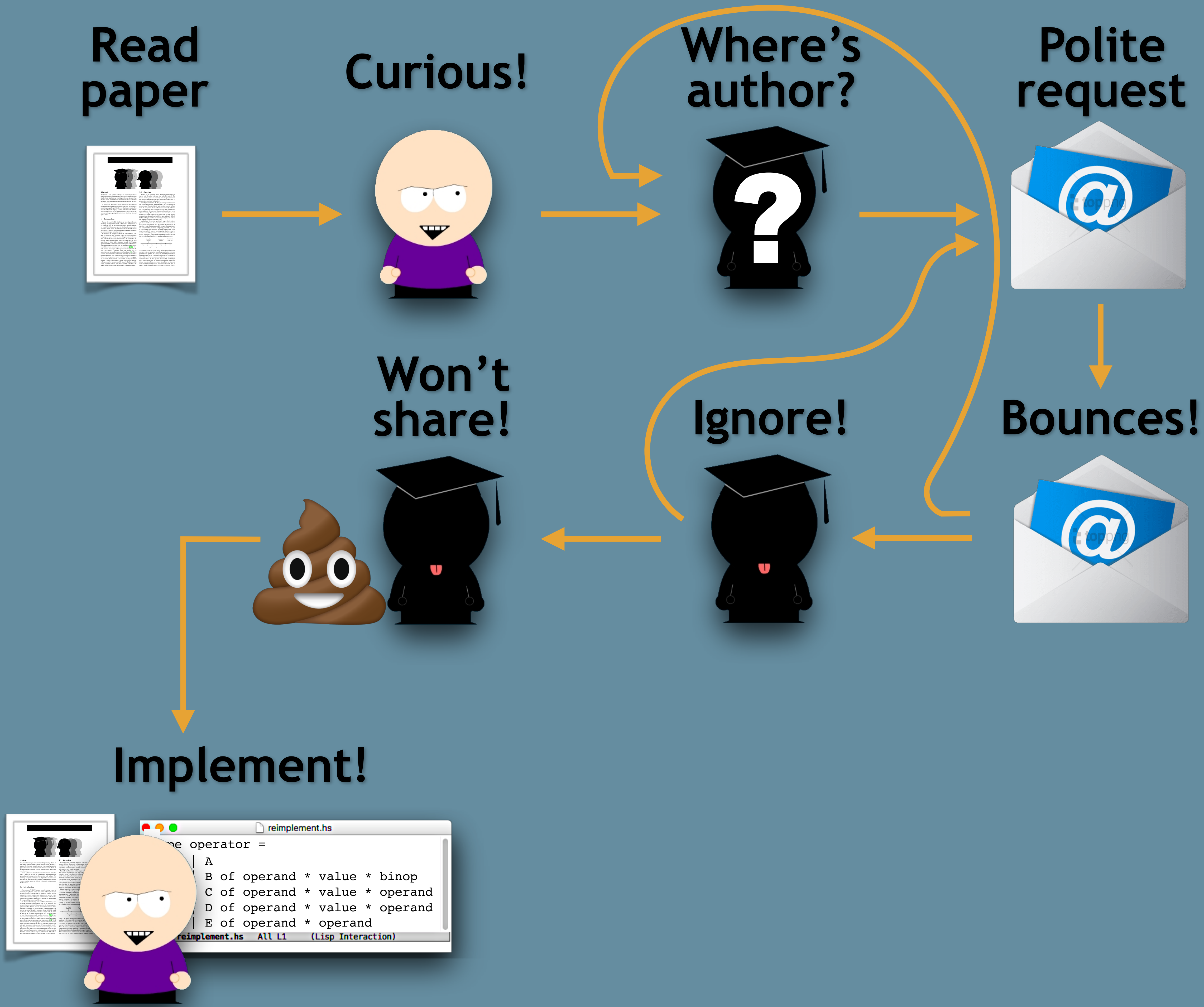

Read paper
Curious!
Where's author?
Polite request
Bounces!


Read paper
Curious!
Where's author?
Polite request
Bounces!
Ignore!

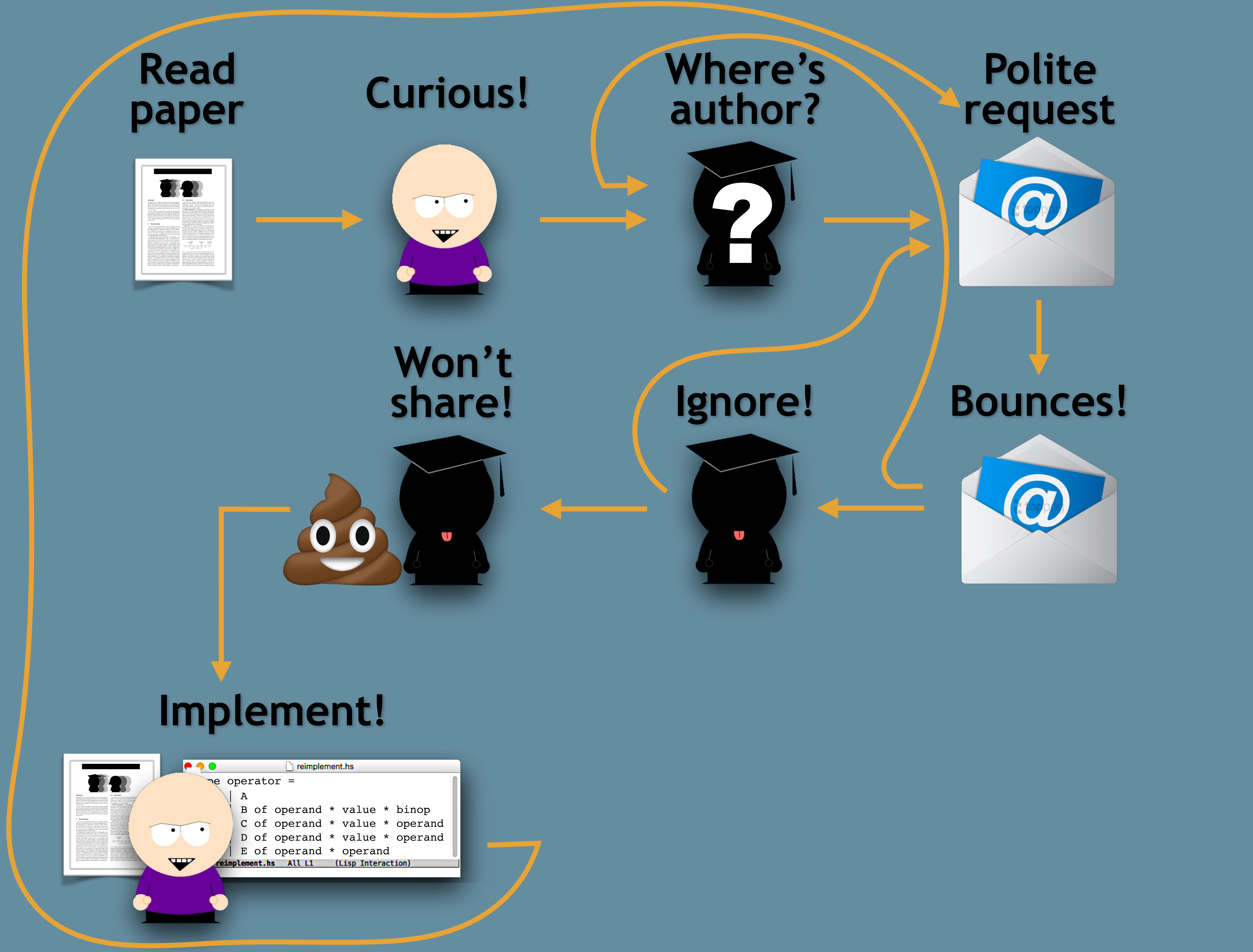

Read paper
Curious!
Where's author?
Polite request
Bounces!
Ignore!


Read paper
Curious!
Where's author?
Polite request
Bounces!
Ignore!
Won't share!

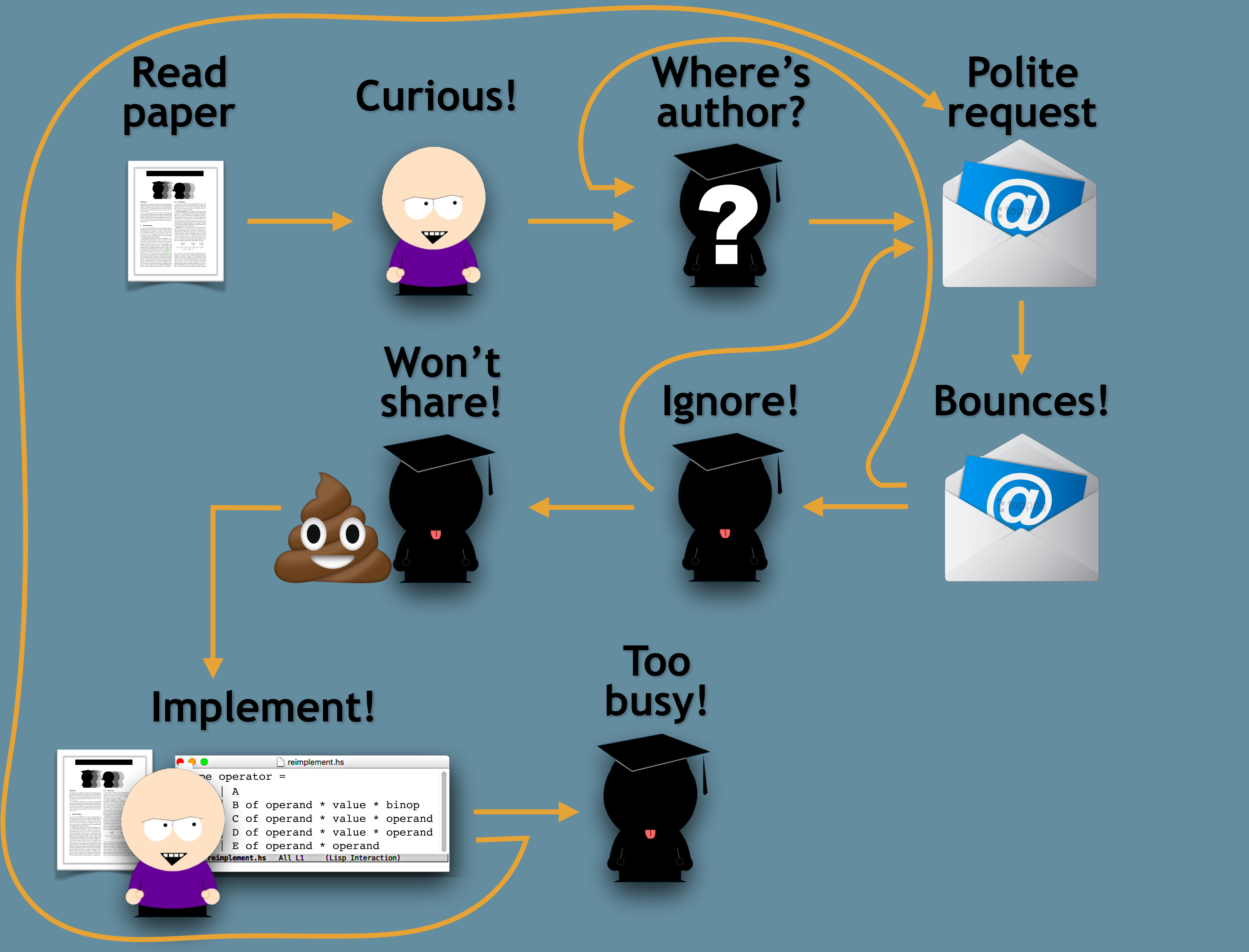

Read paper
Curious!
Where's author?
Polite request
Bounces!
Ignore!
Won't share!



```
pe operator =
  | A
  | B of operand * value * binop
  | C of operand * value * operand
  | D of operand * value * operand
  | E of operand * operand
reimplement.hs   All L1     (Lisp Interaction)
```



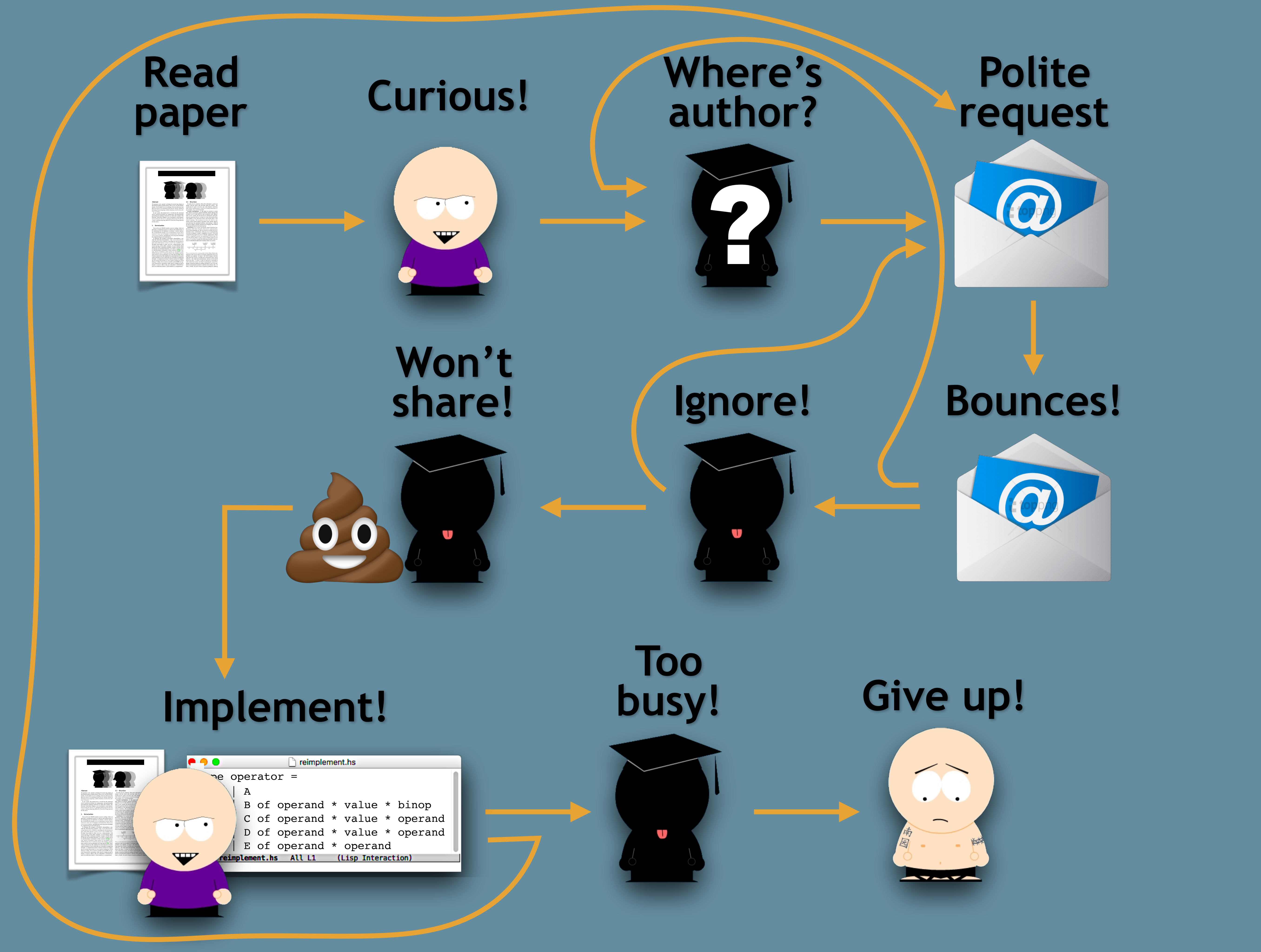
```
pe operator =
  | A
    B of operand * value * binop
    C of operand * value * operand
    D of operand * value * operand
  | E of operand * operand
reimplement.hs   All L1     (Lisp Interaction)
```



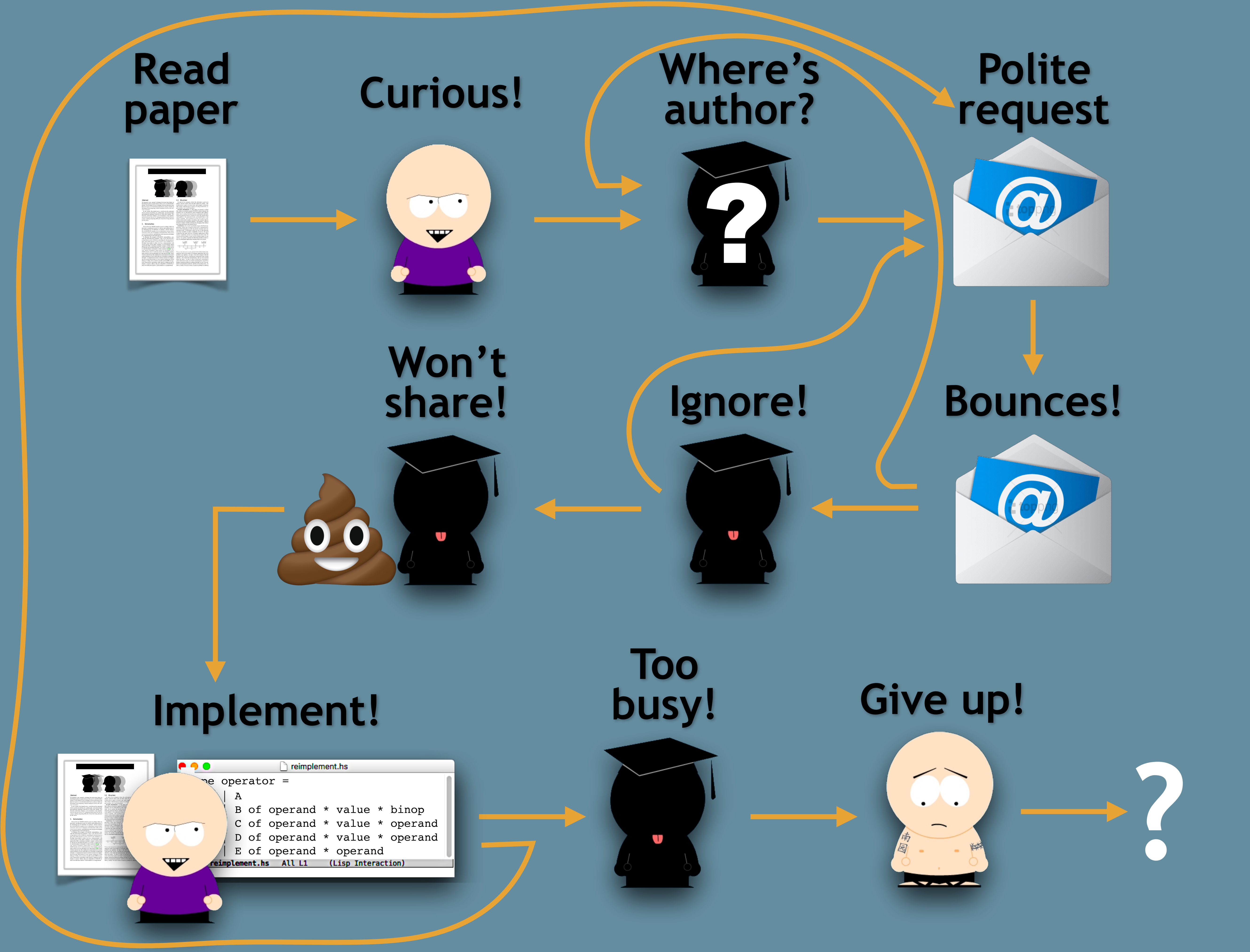
reimplement.hs

```
pe operator =
  | A
  | B of operand * value * binop
  | C of operand * value * operand
  | D of operand * value * operand
  | E of operand * operand
```

```
reimplement.hs   All L1    (Lisp Interaction)
```

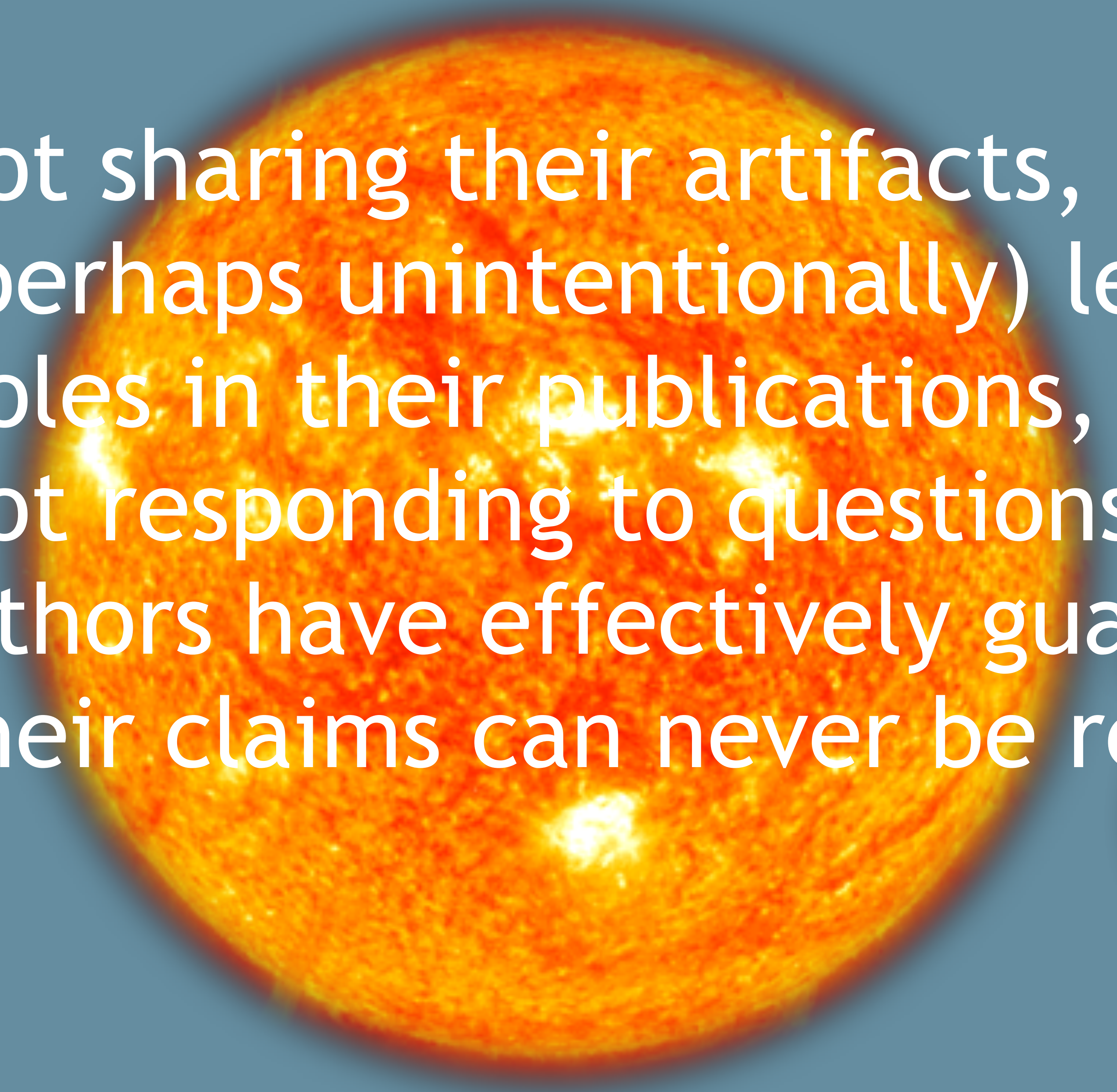


```
pe operator =
 | A
 | B of operand * value * binop
 | C of operand * value * operand
 | D of operand * value * operand
 | E of operand * operand
reimplement.hs   All L1   (Lisp Interaction)
```



```
pe operator =
  | A
  | B of operand * value * binop
  | C of operand * value * operand
  | D of operand * value * operand
  | E of operand * operand
reimplement.hs   All L1     (Lisp Interaction)
```

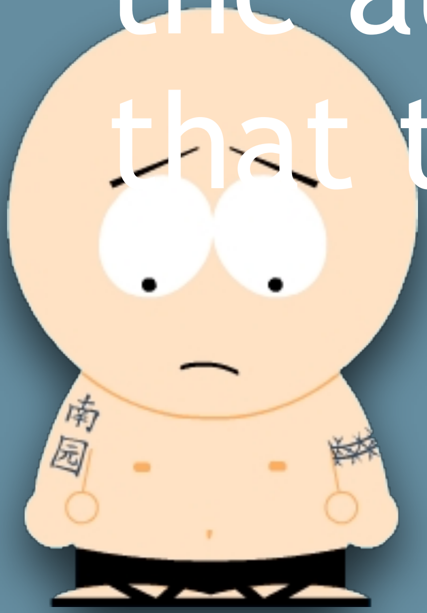
Consequences



By

- not sharing their artifacts,
- (perhaps unintentionally) leaving holes in their publications, and
- not responding to questions,

the authors have effectively guaranteed that their claims can never be refuted.



Consequences

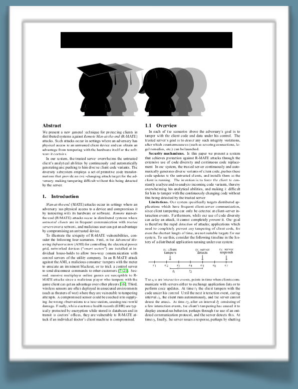
By

- not sharing their artifacts,
- (perhaps unintentionally) leaving holes in their publications, and
- not responding to questions,

the authors have effectively guaranteed that their claims can never be refuted.

8th Law of Artifact Sharing
(Pretschner’s Law)

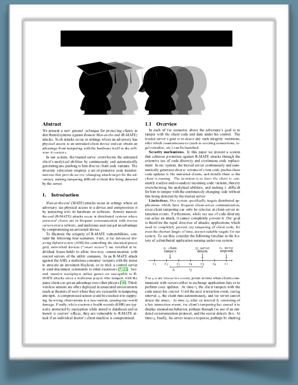
The probability of getting code out of someone is inversely proportional to the outrageousness of the claims in the paper.



Research Artifacts

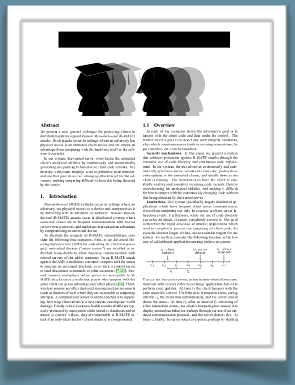
- Code
- Data
- Experiments
- ...

Repeatability



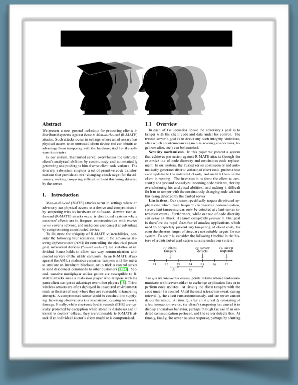
Research Artifacts

- Code
- Data
- Experiments
- ...



Verify results

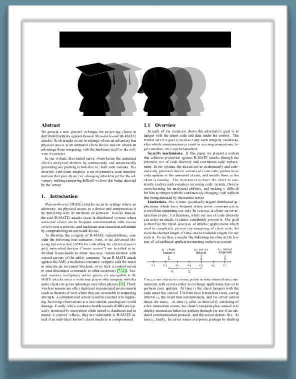
Research Artifacts



Research Artifacts

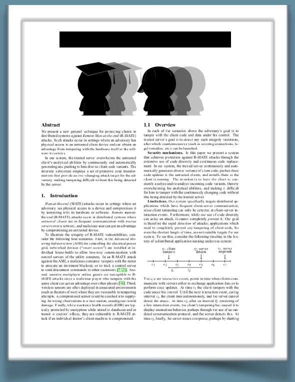
- Code
- Data
- Experiments
- ...

Repeatability

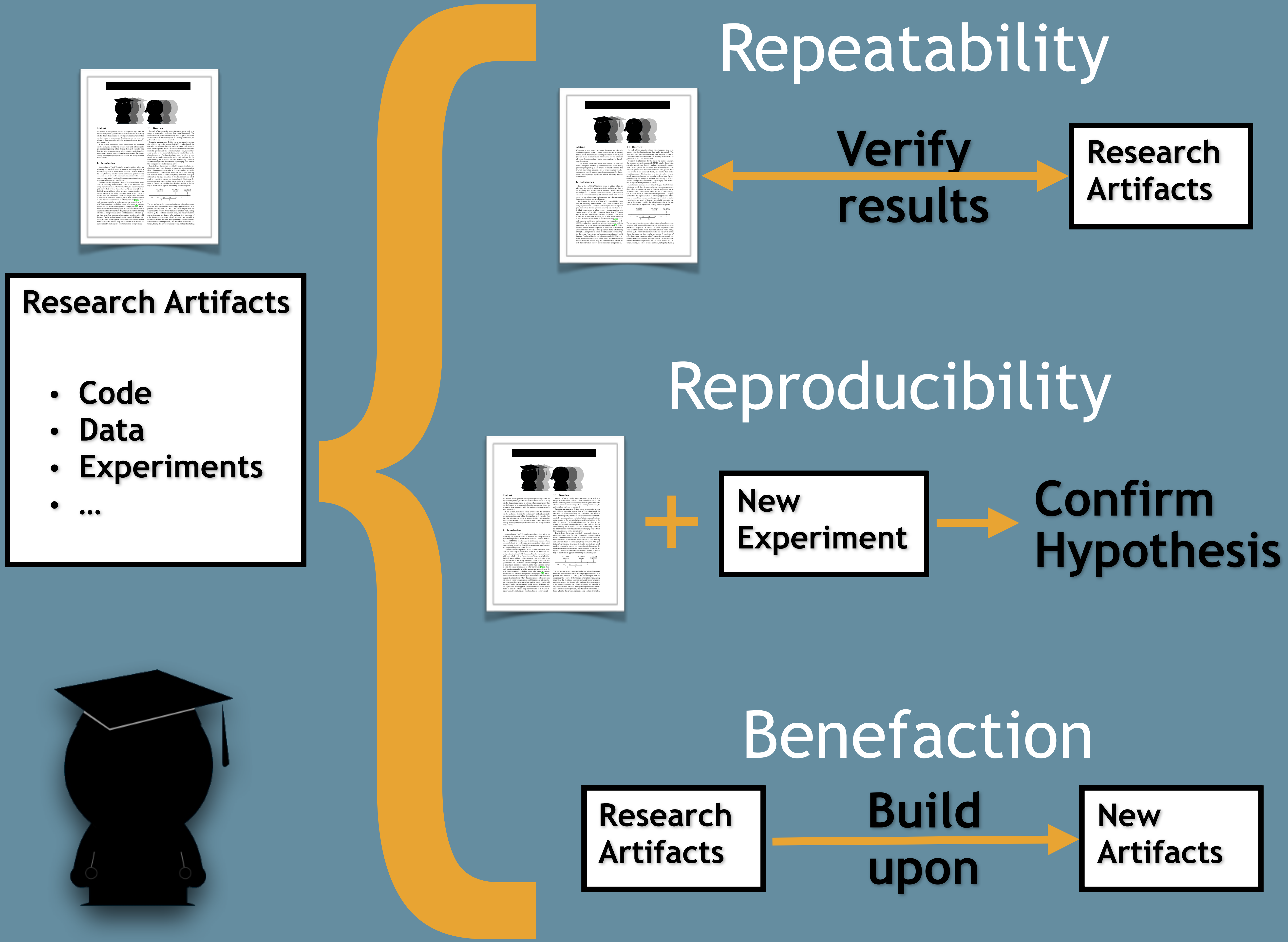


Verify results ⟷ Research Artifacts

Reproducibility

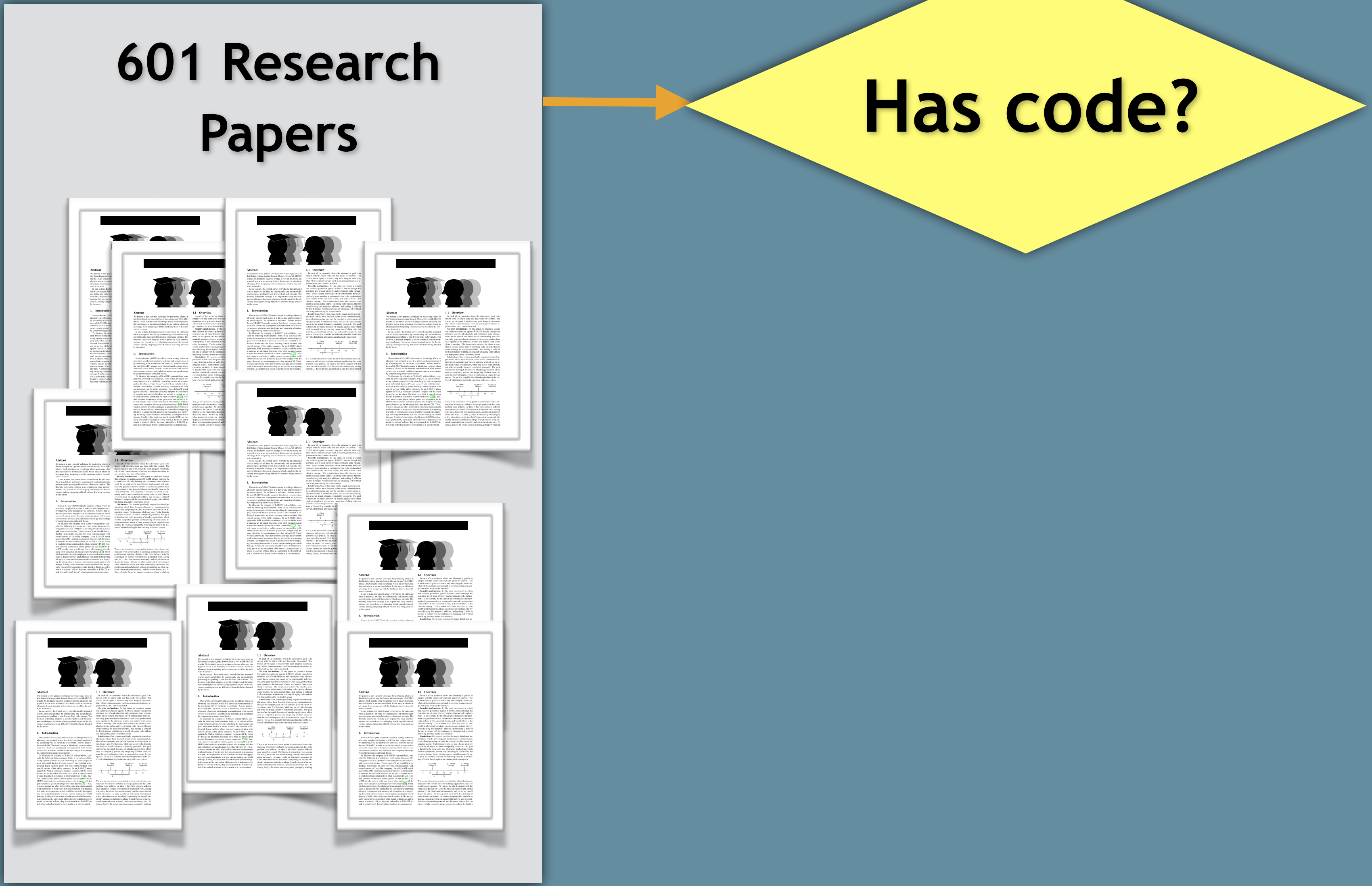


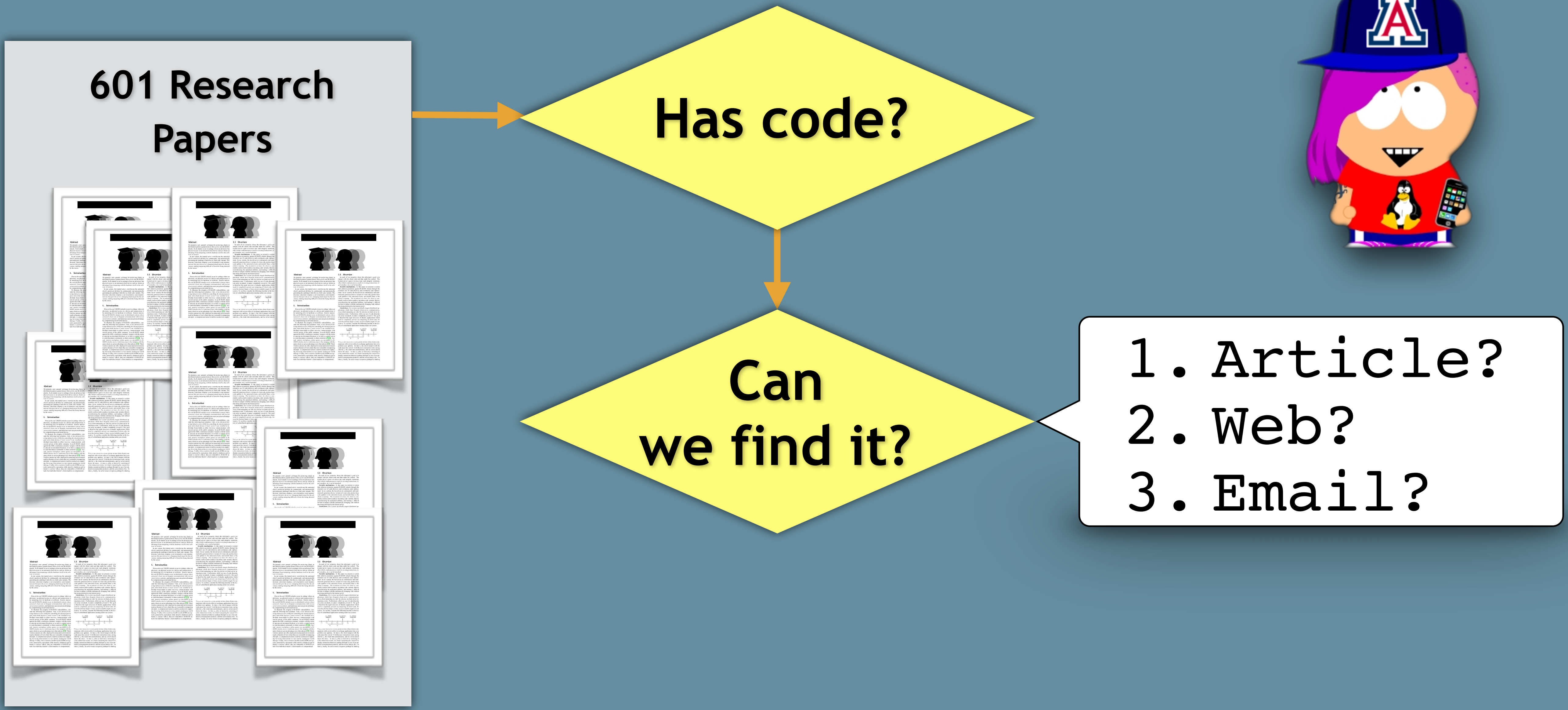
+ New Experiment → Confirm Hypothesis

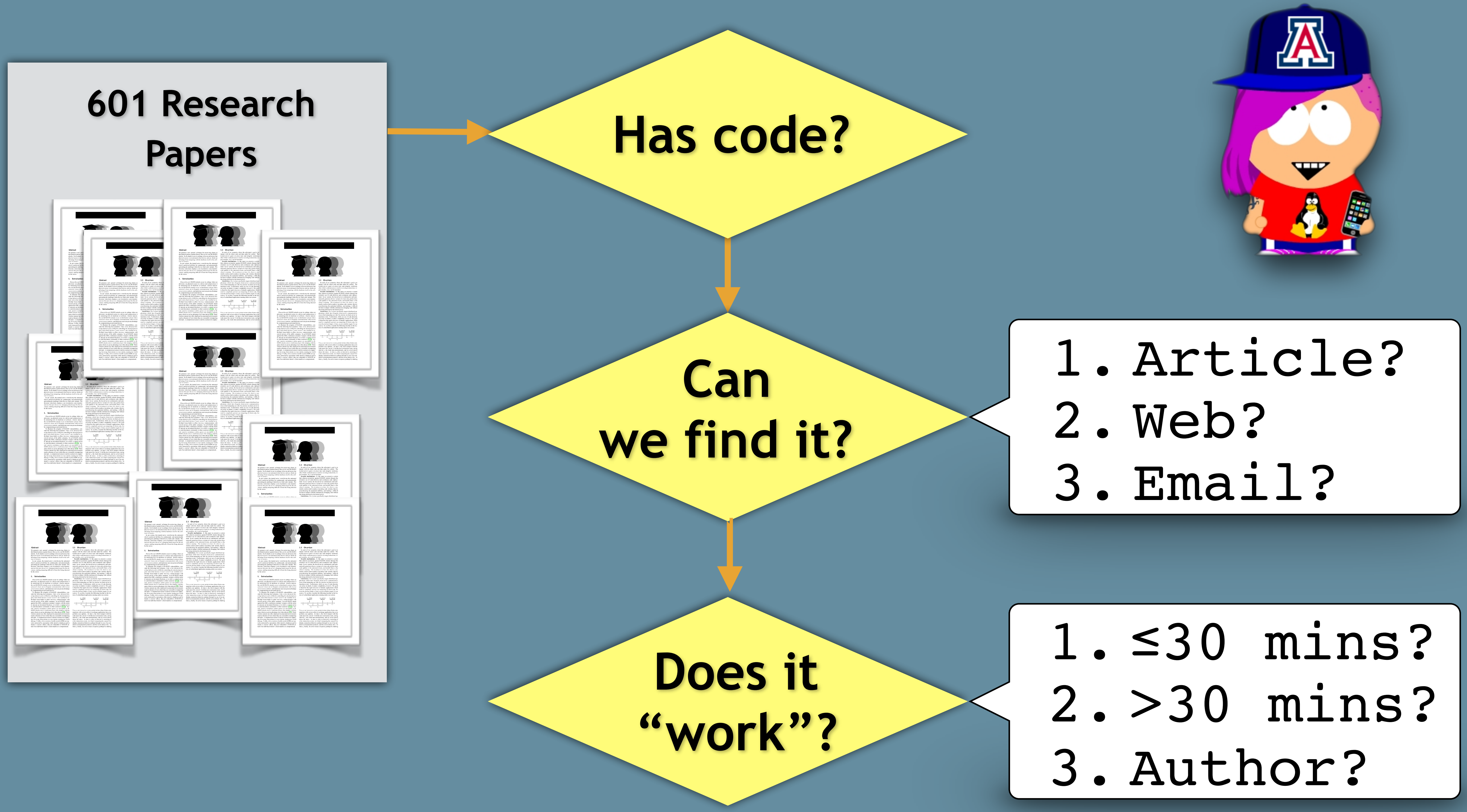

Research Artifacts
• Code
• Data
• Experiments
• …
Repeatability
Verify results
Research Artifacts
Reproducibility
+
New Experiment
Confirm Hypothesis
Benefaction
Research Artifacts
Build upon
New Artifacts

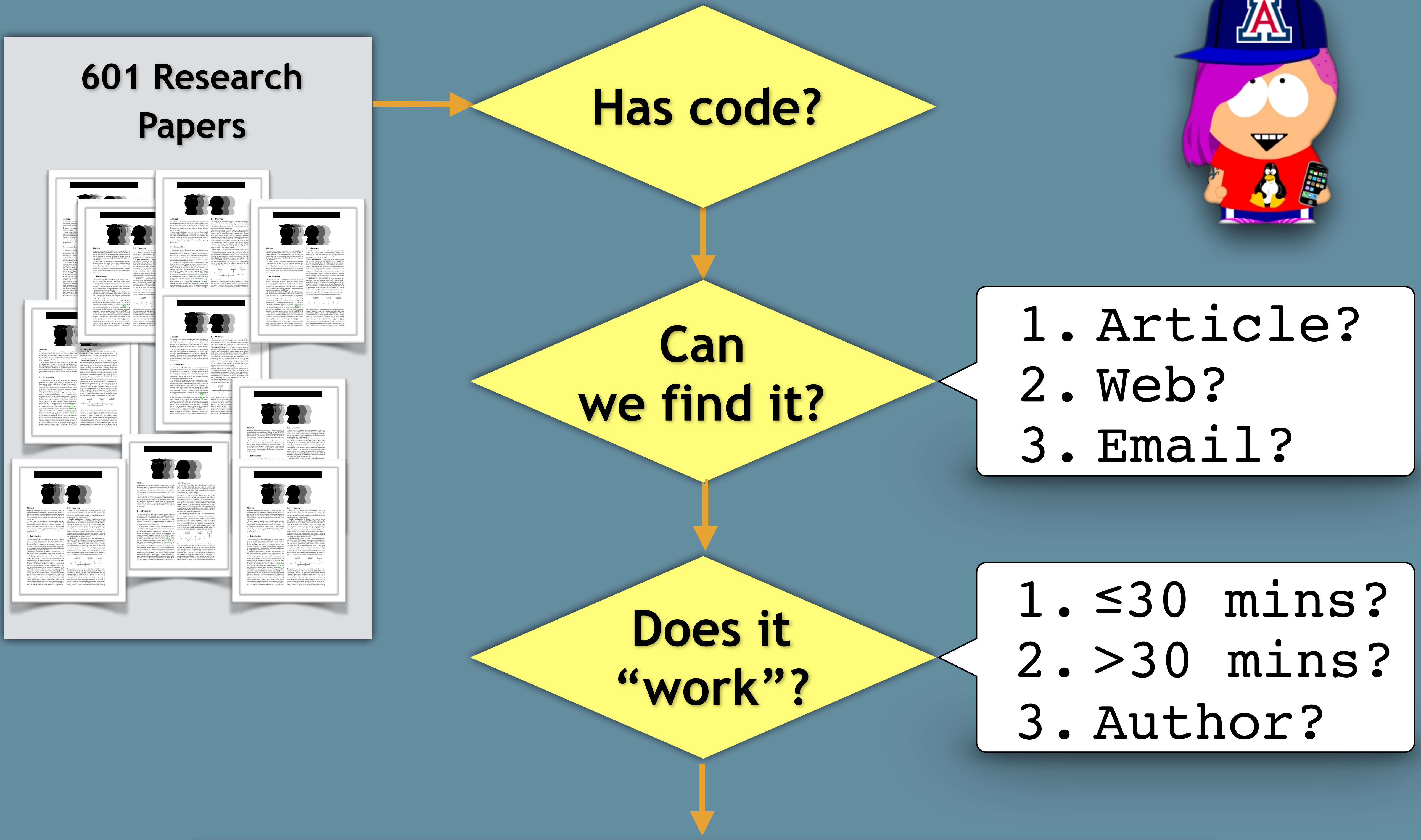
The Deception Study

601 Research Papers


601 Research Papers
Has code?

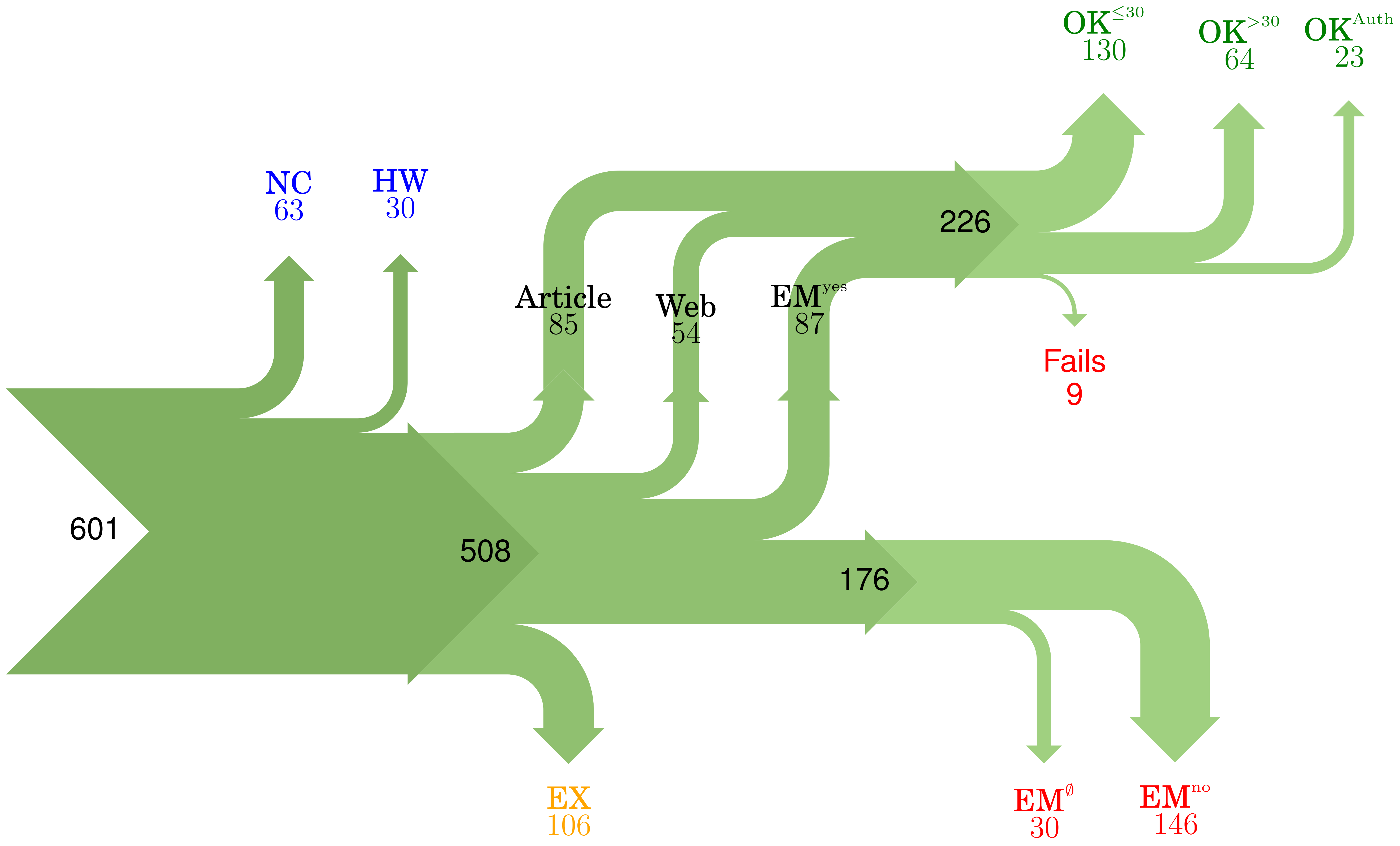

601 Research Papers
Has code?
Can we find it?
1. Article?
2. Web?
3. Email?


601 Research Papers
Has code?
Can we find it?
1. Article?
2. Web?
3. Email?
Does it "work"?
1. ≤30 mins?
2. >30 mins?
3. Author?

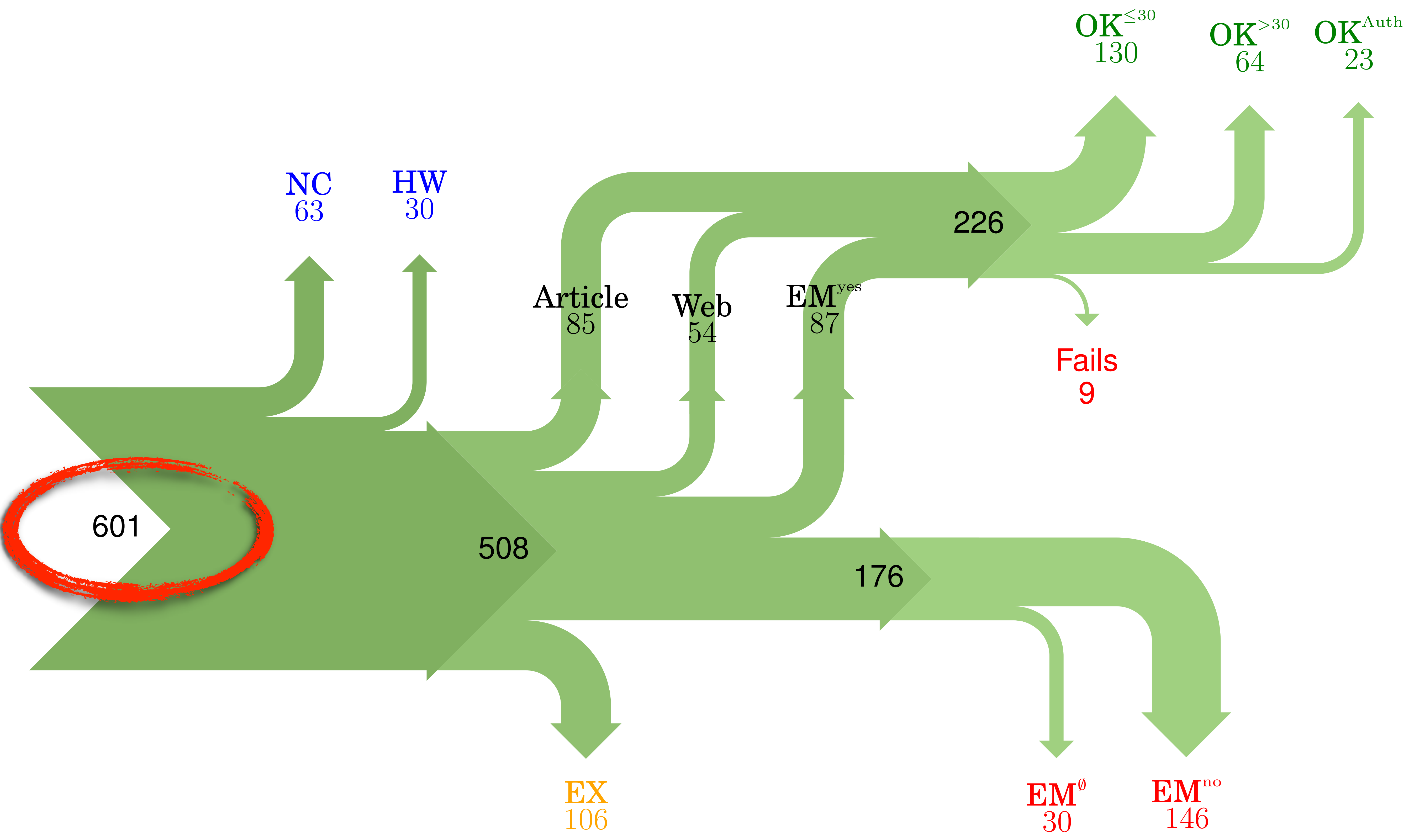


Weakly Repeatable

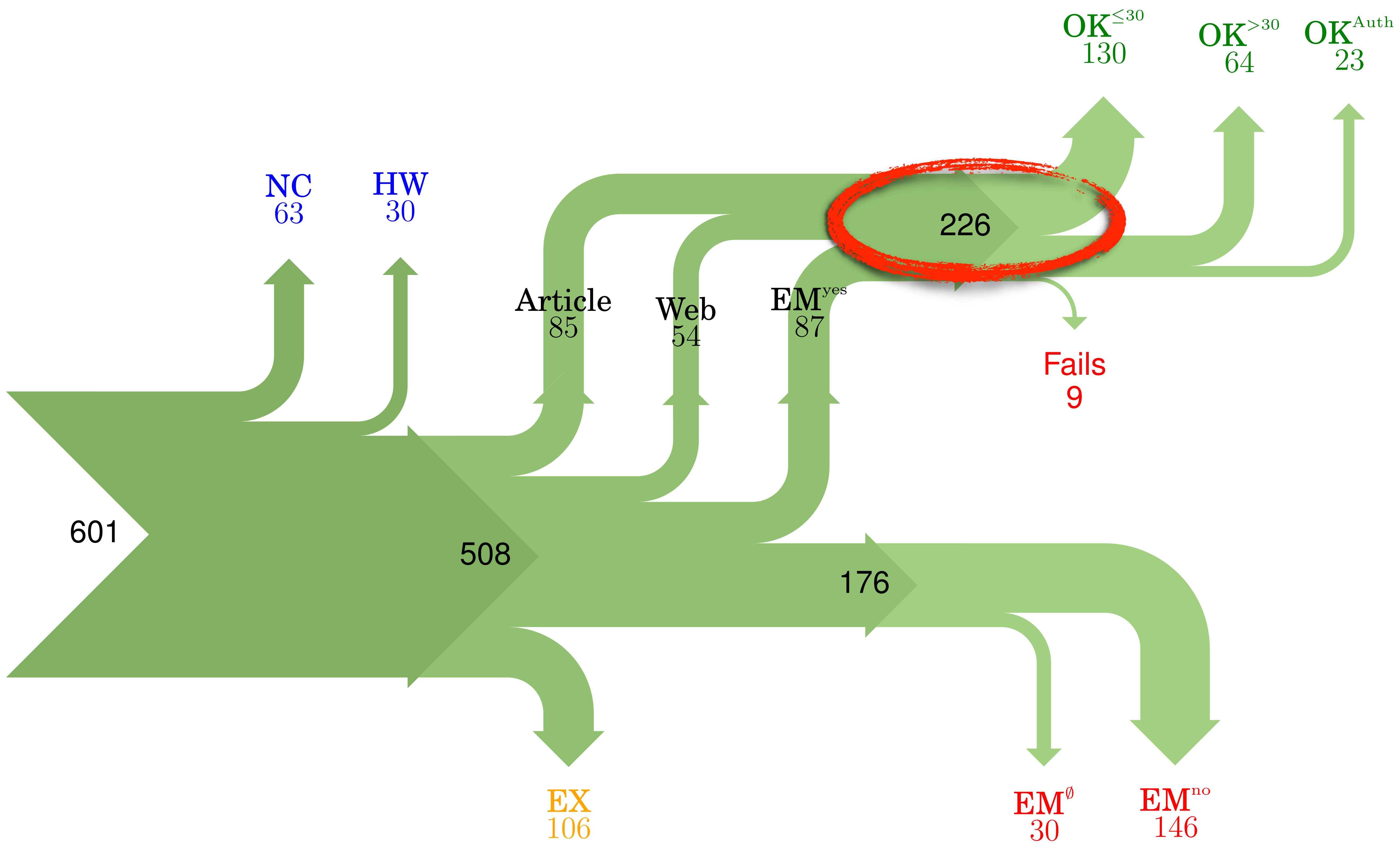
Authors share their code, and it builds.

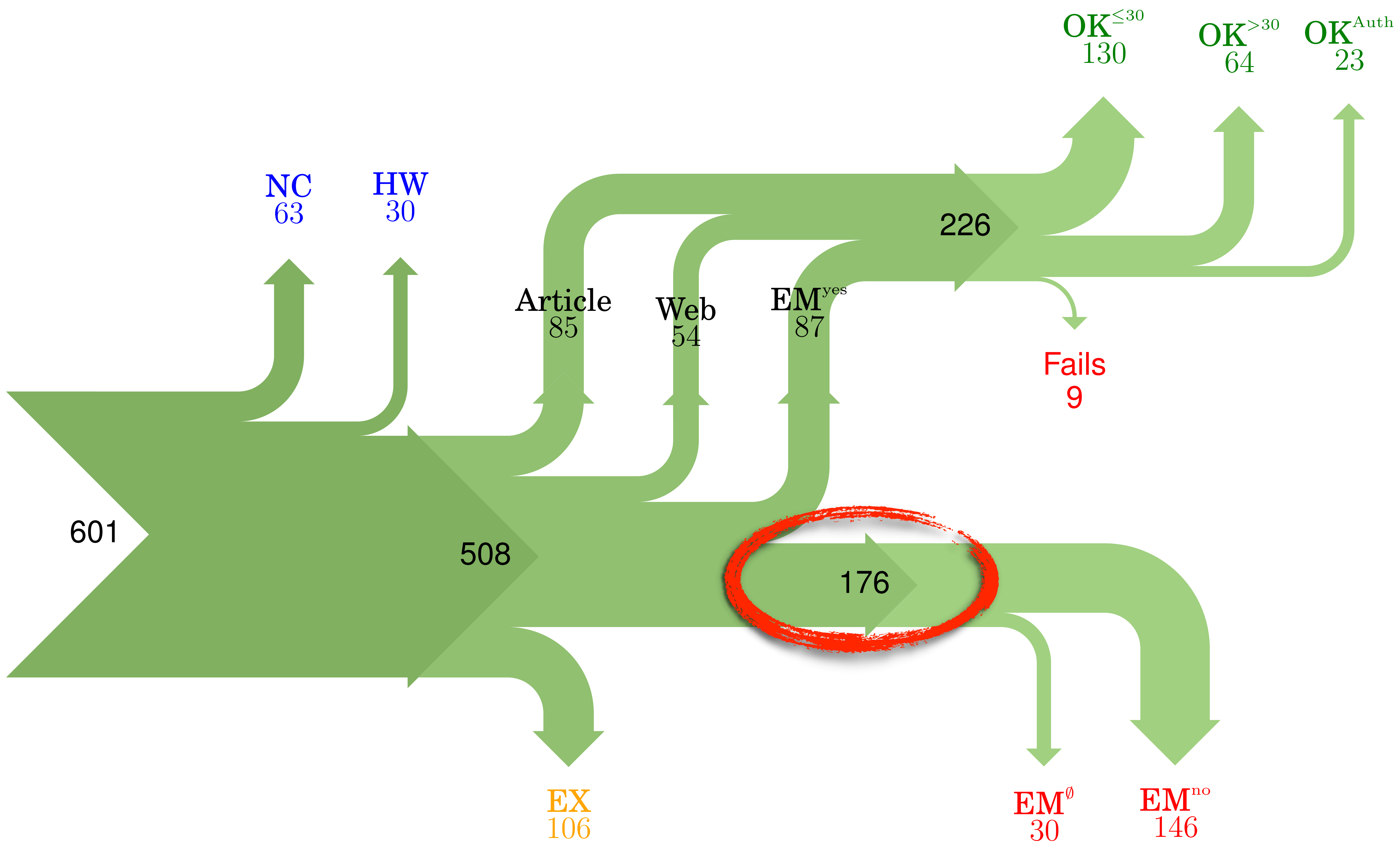


601
NC 63
HW 30
508
Article 85
Web 54
EM^{yes} 87
226
OK$^{\leq 30}$ 130
OK$^{>30}$ 64
OK^{Auth} 23
Fails 9
176
EX 106
EM$^{\emptyset}$ 30
EM^{no} 146



601
NC 63
HW 30
508
EX 106
Article 85
Web 54
EM^{yes} 87
226
OK$^{\leq 30}$ 130
OK$^{>30}$ 64
OK^{Auth} 23
Fails 9
176
EM$^{\emptyset}$ 30
EM^{no} 146


OK≤30 130
OK>30 64
OKAuth 23
NC 63
HW 30
226
Article 85
Web 54
EMyes 87
Fails 9
601
508
176
EX 106
EMØ 30
EMno 146



OK$^{\leq 30}$ 130
OK$^{>30}$ 64
OK^{Auth} 23
NC 63
HW 30
226
Article 85
Web 54
EM^{yes} 87
Fails 9
601
508
176
EX 106
EM$^{\emptyset}$ 30
EM^{no} 146

The good news … I was able to find some code. I am just **hoping** that it … **matches the implementation** we … used for the paper.

Versioning

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

Available Soon

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

Personnel Issues

[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

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

Lost Code

The code … is … **hardly usable by anyone** other than the authors … due to our decision to use [obscure variant of obscure language]

Design Issues

7th Law of Artifact Sharing
(Prepare to Share)

Unless a project starts with the express goal of post-publication artifact sharing, getting the right code, in a timely fashion, out of the project is virtually impossible.

We will not provide the software … [because we spent] **more time getting outsiders up to speed than on our own research**.

Academic Tradeoffs

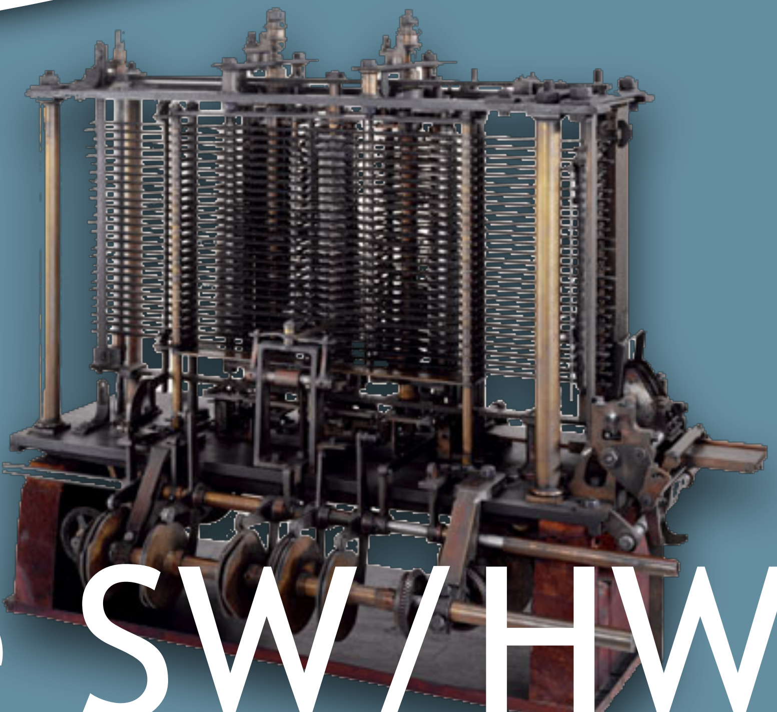
We will not provide the software … [because we spent] **more time getting outsiders up to speed than on our own research**.

Academic Tradeoffs

... **we can't share what did for this paper**. ... this is not in the academic tradition, but this is a hazard in an industrial lab.

Industrial Lab Tradeoffs

We have no plans to make the scheduler's source code publicly available … because [ancient OS] as such **does not exist anymore**.

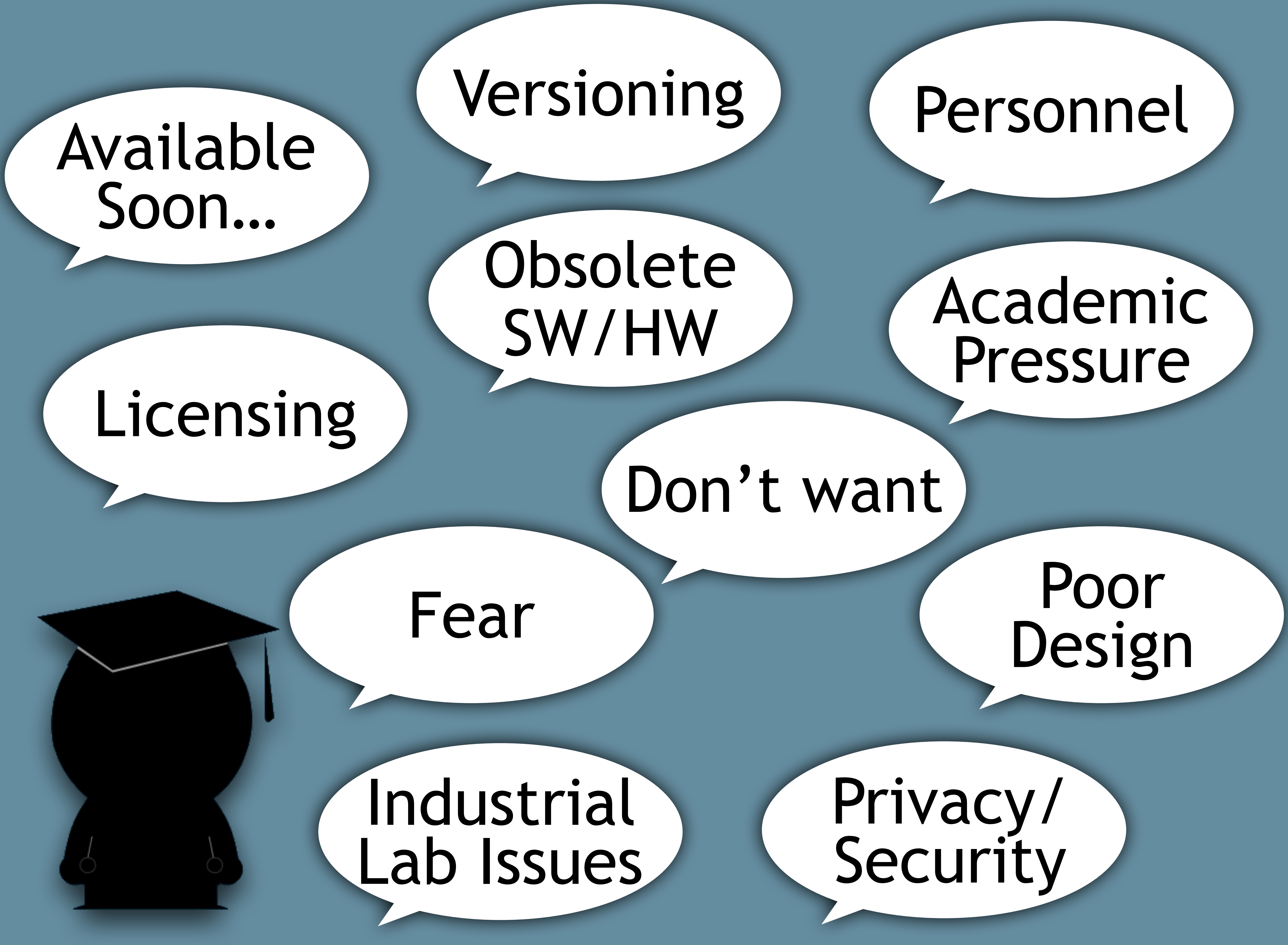


Obsolete SW/HW

We have an agreement with the [business], and we cannot release the code because of the potential **privacy risks** …

Privacy/Security

Fear


Available Soon...
Versioning
Personnel
Obsolete SW/HW
Academic Pressure
Licensing
Don’t want
Fear
Poor Design
Industrial Lab Issues
Privacy/ Security

Sharing Proposal — #1 —

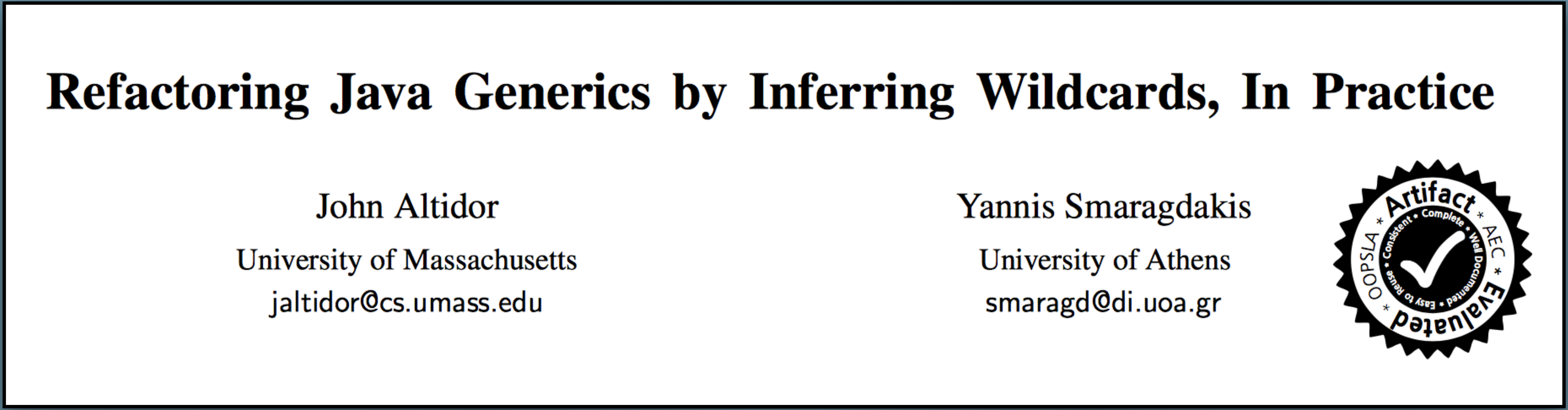
Artifact Curation

Sharing Proposal — #1 —

Artifact Curation

TECHNO-FIX

ACM Artifact Curation

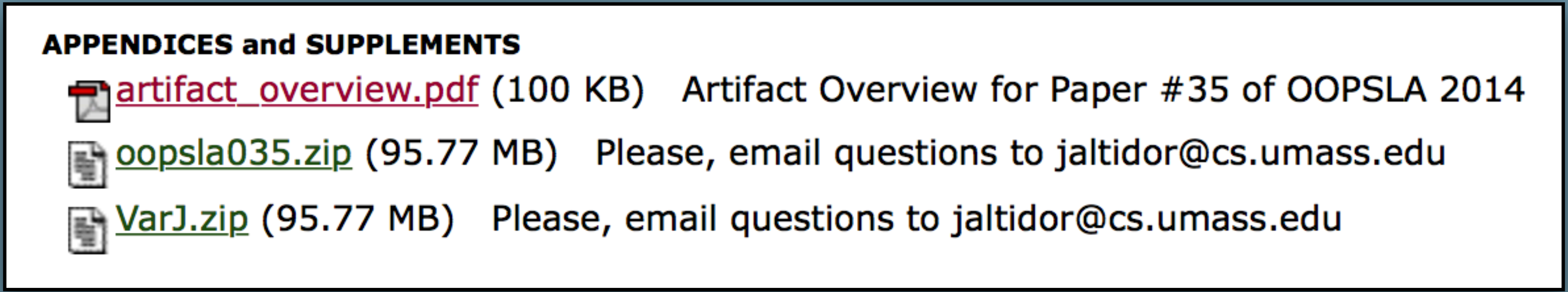

Artifact
Complete
Well Documented
Easy to Reuse
Consistent
OOPSLA
AEC
Evaluated

Refactoring Java Generics by Inferring Wildcards, In Practice

John Altidor
University of Massachusetts
jaltidor@cs.umass.edu

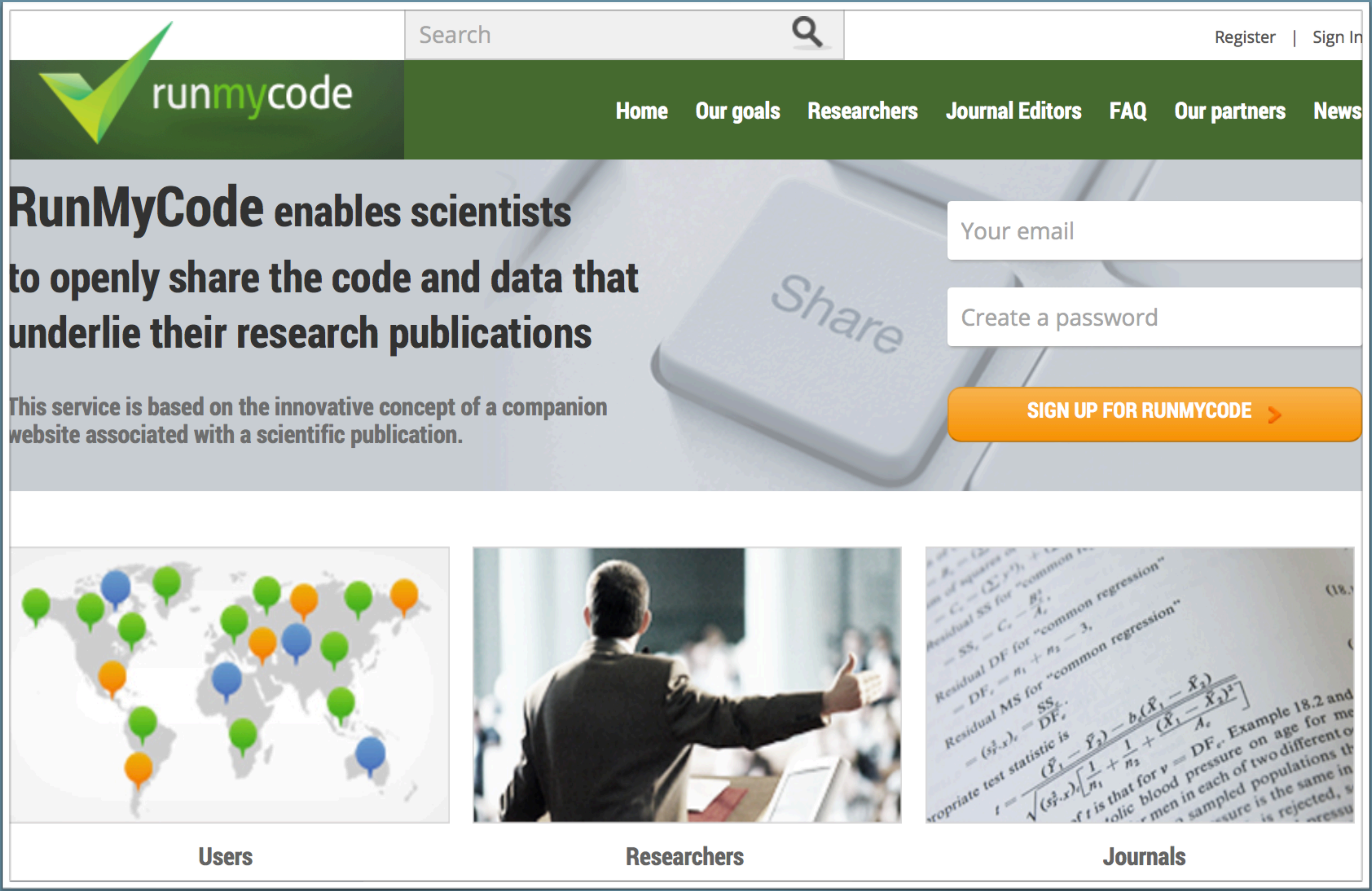
Yannis Smaragdakis
University of Athens
smaragd@di.uoa.gr

ACM DL DIGITAL LIBRARY



APPENDICES and SUPPLEMENTS

- artifact_overview.pdf (100 KB) Artifact Overview for Paper #35 of OOPSLA 2014
- oopsla035.zip (95.77 MB) Please, email questions to jaltidor@cs.umass.edu
- VarJ.zip (95.77 MB) Please, email questions to jaltidor@cs.umass.edu



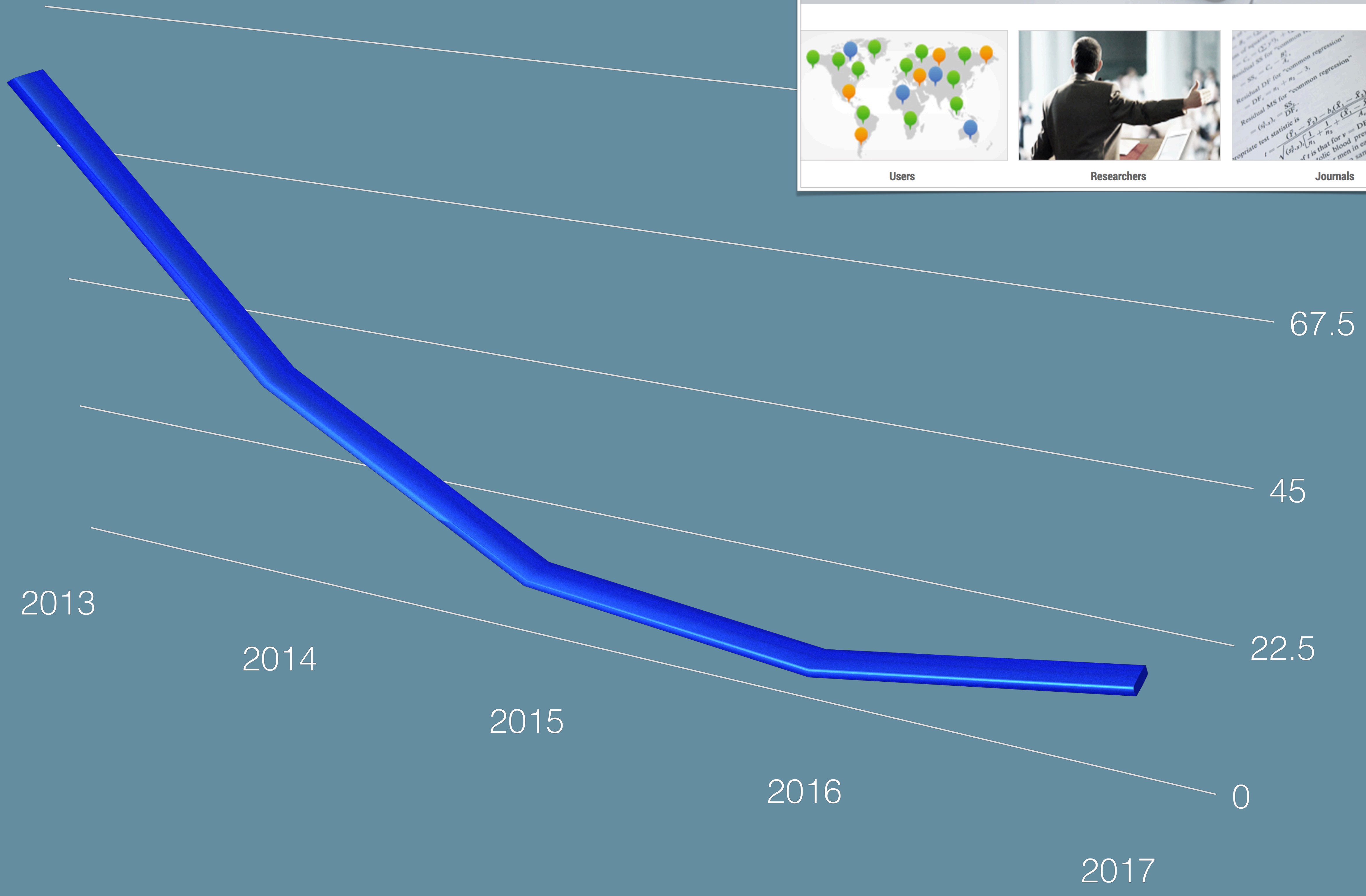
RunMyCode enables scientists to openly share the code and data that underlie their research publications

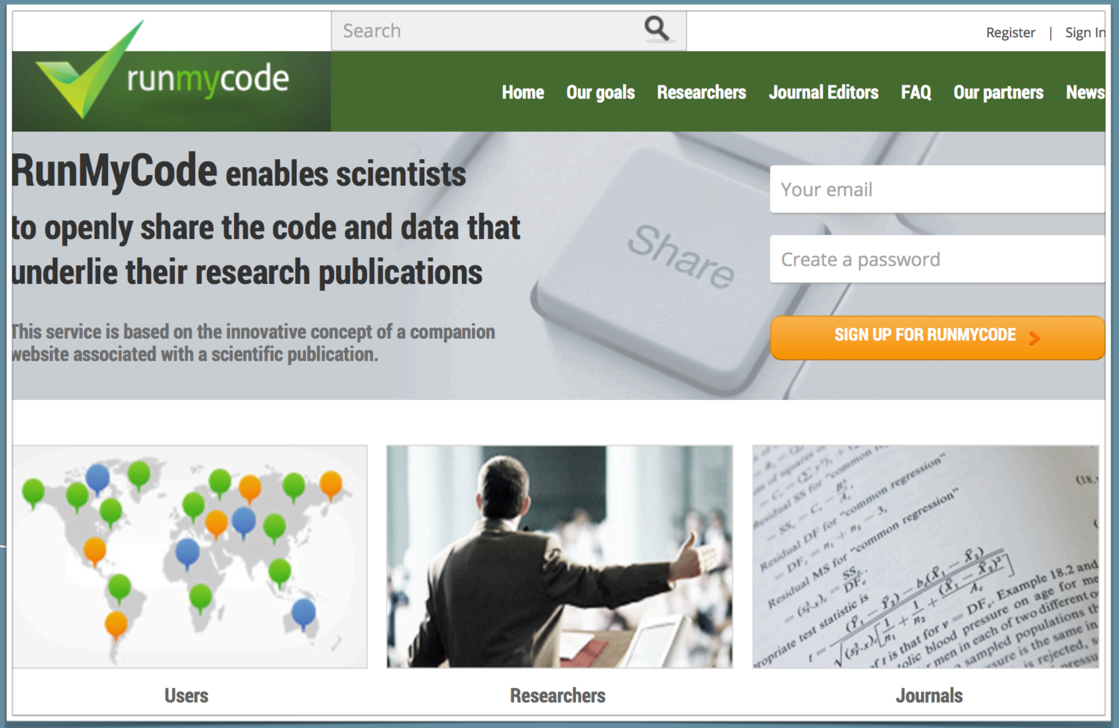
This service is based on the innovative concept of a companion website associated with a scientific publication.

Users

Researchers

Journals

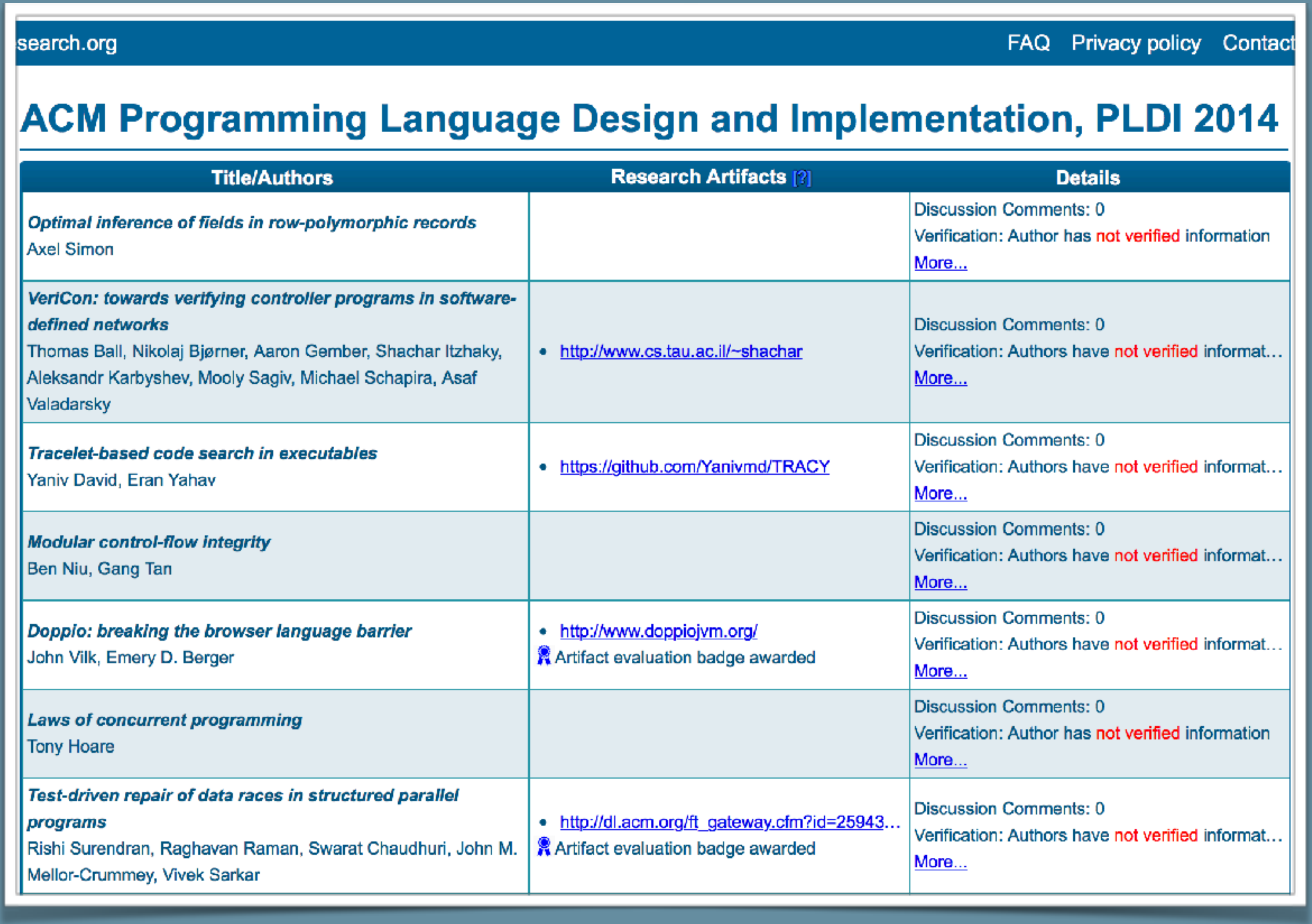




RunMyCode enables scientists to openly share the code and data that underlie their research publications

This service is based on the innovative concept of a companion website associated with a scientific publication.

FindResearch.org



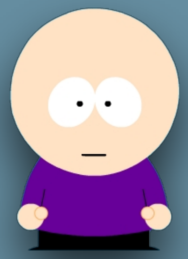
search.org FAQ Privacy policy Contact

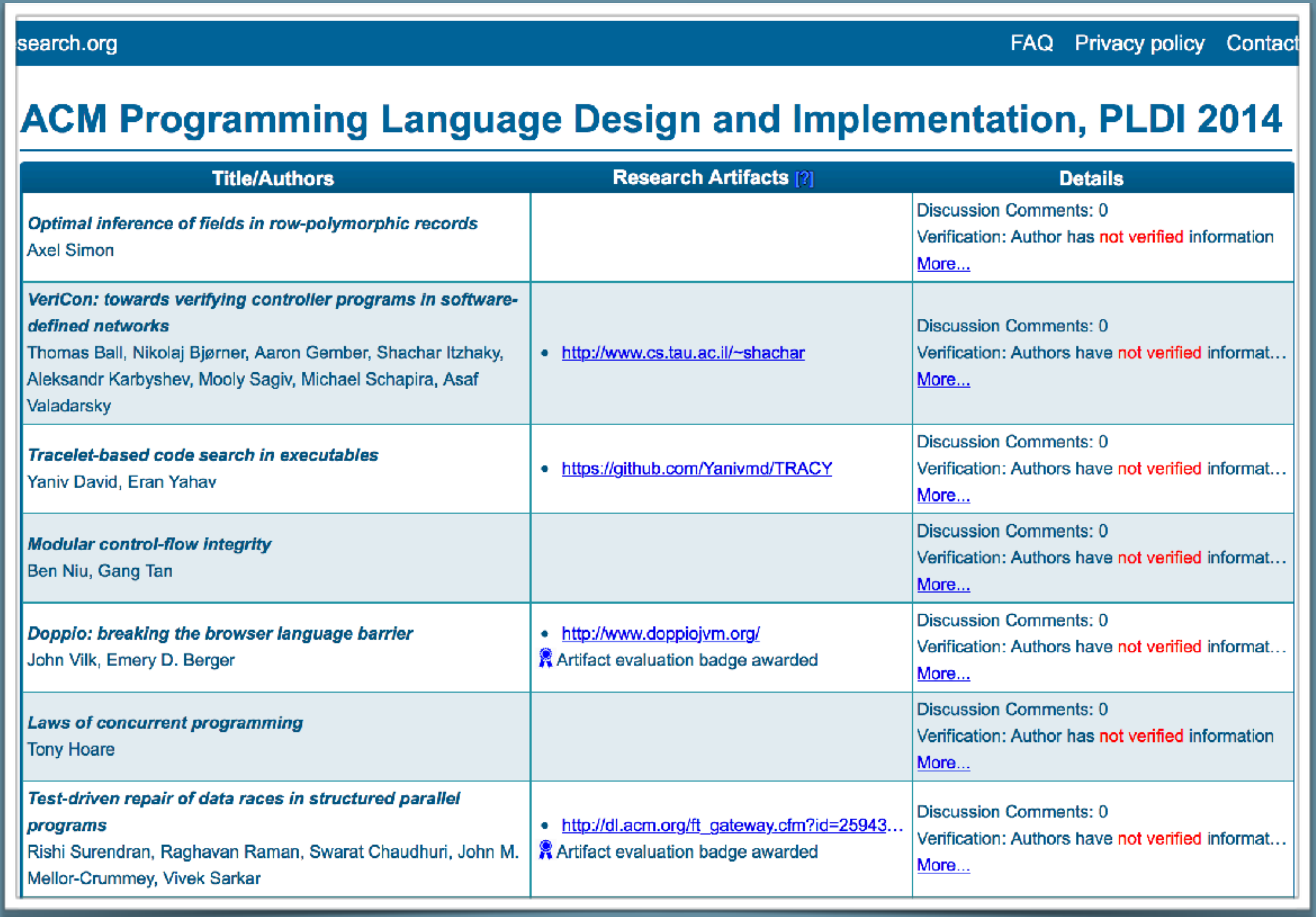
ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat… More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat… More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat… More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat… More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943… Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat… More...

1. Help the public find artifacts
2. Motivate researchers to share

FindResearch.org



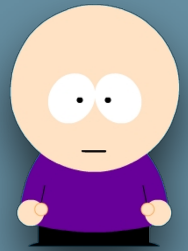


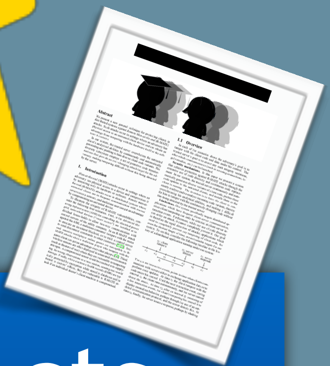
search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

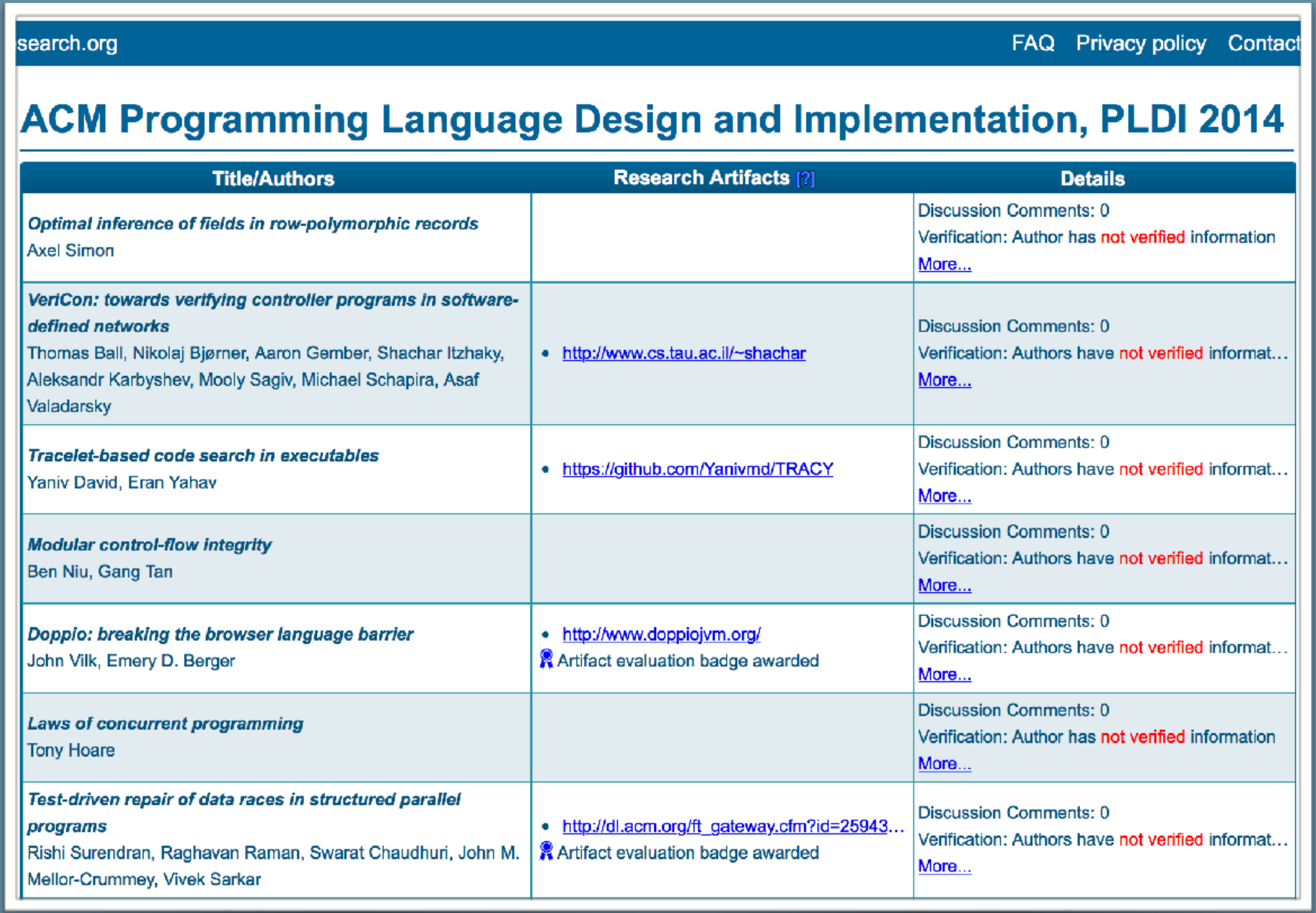
Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

FindResearch.org





Find Artifacts
Emails, Grants

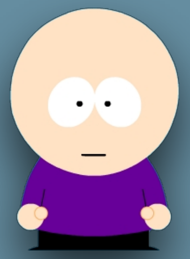


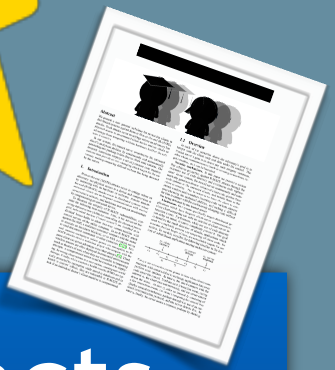
search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

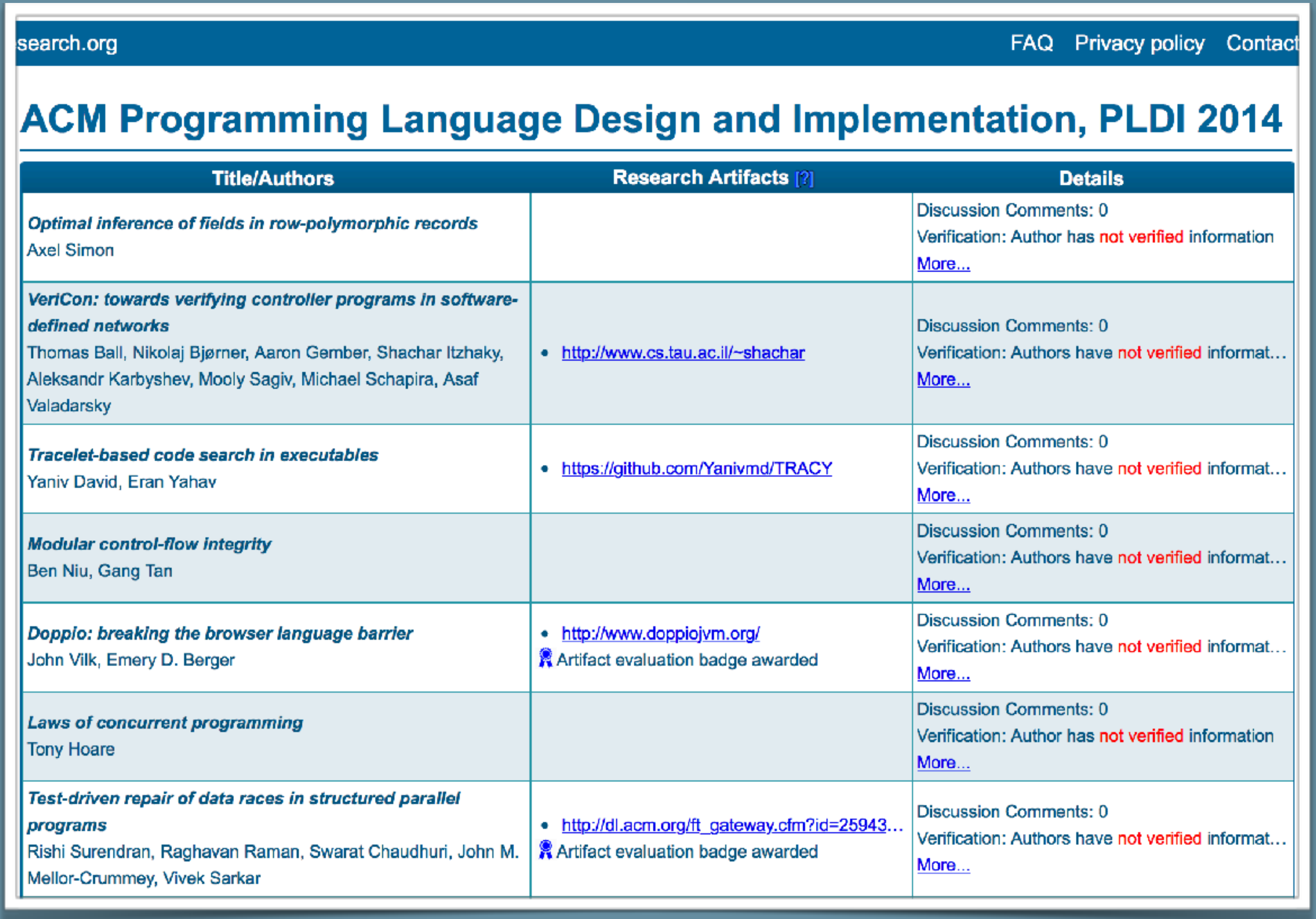
Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

FindResearch.org





- Find Artifacts Emails, Grants
- Verify Information



search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

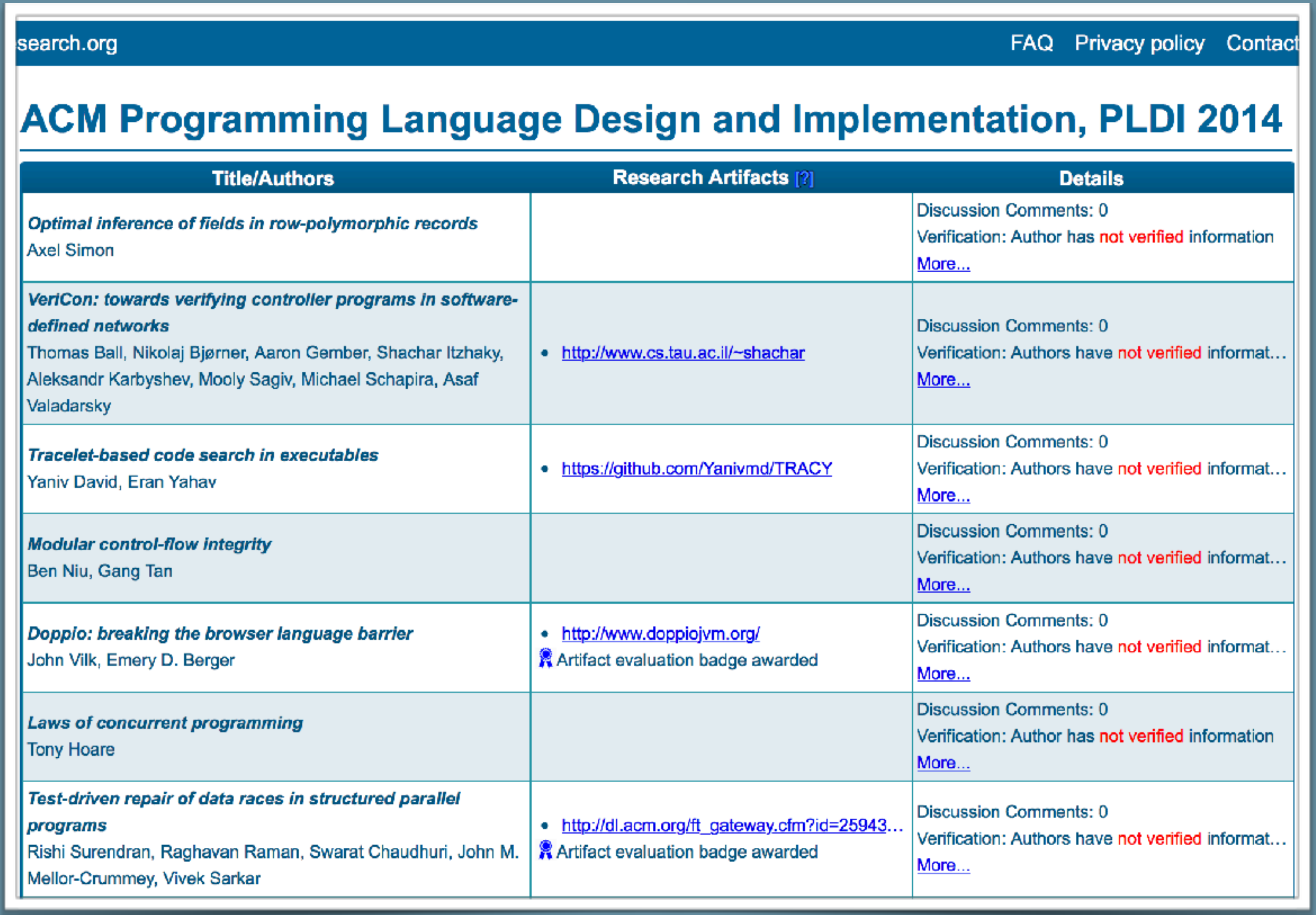
Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

FindResearch.org

Find Artifacts Emails, Grants

Verify Information

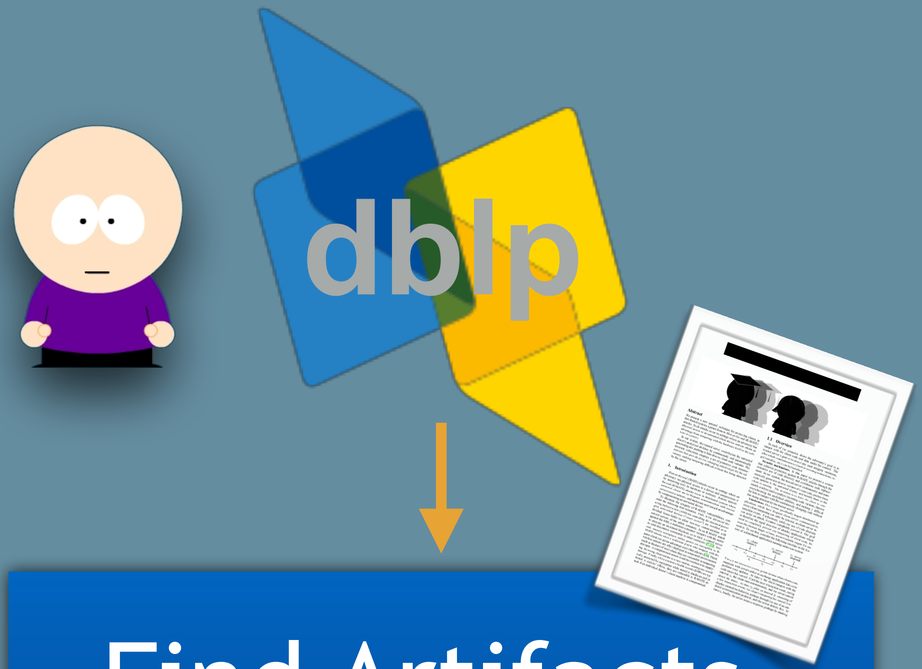
Publish


search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

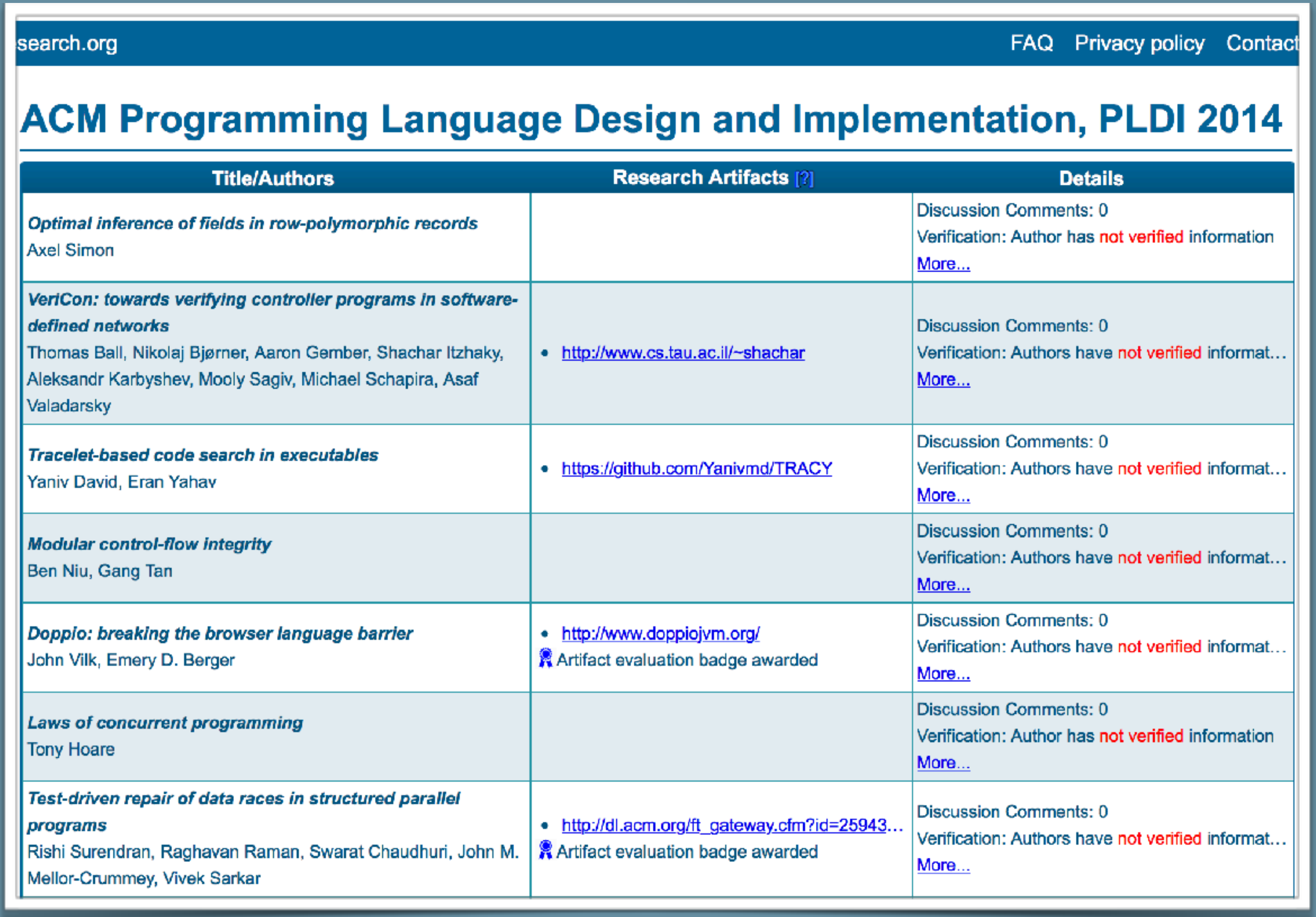
FindResearch.org



Find Artifacts
Emails, Grants

Verify
Information

Publish



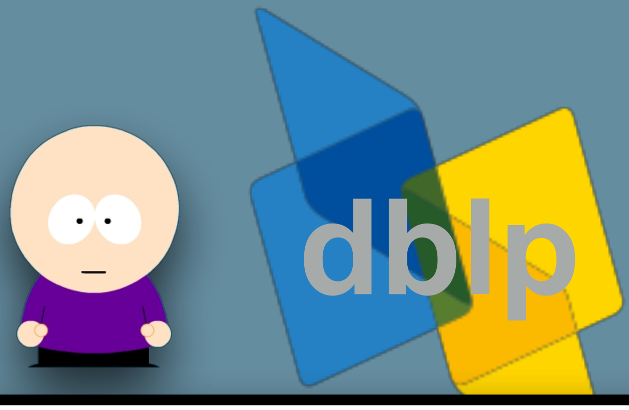
search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

Title/Authors	Research Artifacts [?]	Details
Optimal inference of fields in row-polymorphic records Axel Simon		Discussion Comments: 0 Verification: Author has not verified information More...
VeriCon: towards verifying controller programs in software-defined networks Thomas Ball, Nikolaj Bjørner, Aaron Gember, Shachar Itzhaky, Aleksandr Karbyshev, Mooly Sagiv, Michael Schapira, Asaf Valadarsky	• http://www.cs.tau.ac.il/~shachar	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Tracelet-based code search in executables Yaniv David, Eran Yahav	• https://github.com/Yanivmd/TRACY	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Modular control-flow integrity Ben Niu, Gang Tan		Discussion Comments: 0 Verification: Authors have not verified informat... More...
Doppio: breaking the browser language barrier John Vilk, Emery D. Berger	• http://www.doppiojvm.org/ Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...
Laws of concurrent programming Tony Hoare		Discussion Comments: 0 Verification: Author has not verified information More...
Test-driven repair of data races in structured parallel programs Rishi Surendran, Raghavan Raman, Swarat Chaudhuri, John M. Mellor-Crummey, Vivek Sarkar	• http://dl.acm.org/ft_gateway.cfm?id=25943... Artifact evaluation badge awarded	Discussion Comments: 0 Verification: Authors have not verified informat... More...

Discuss!

FindResearch.org



ACM Programming Language Design and Implementation, PLDI 2014

- 225 conferences
- 18,000 articles
- 39,000 unique authors
- 64,000 verification emails sent

FindResearch.org

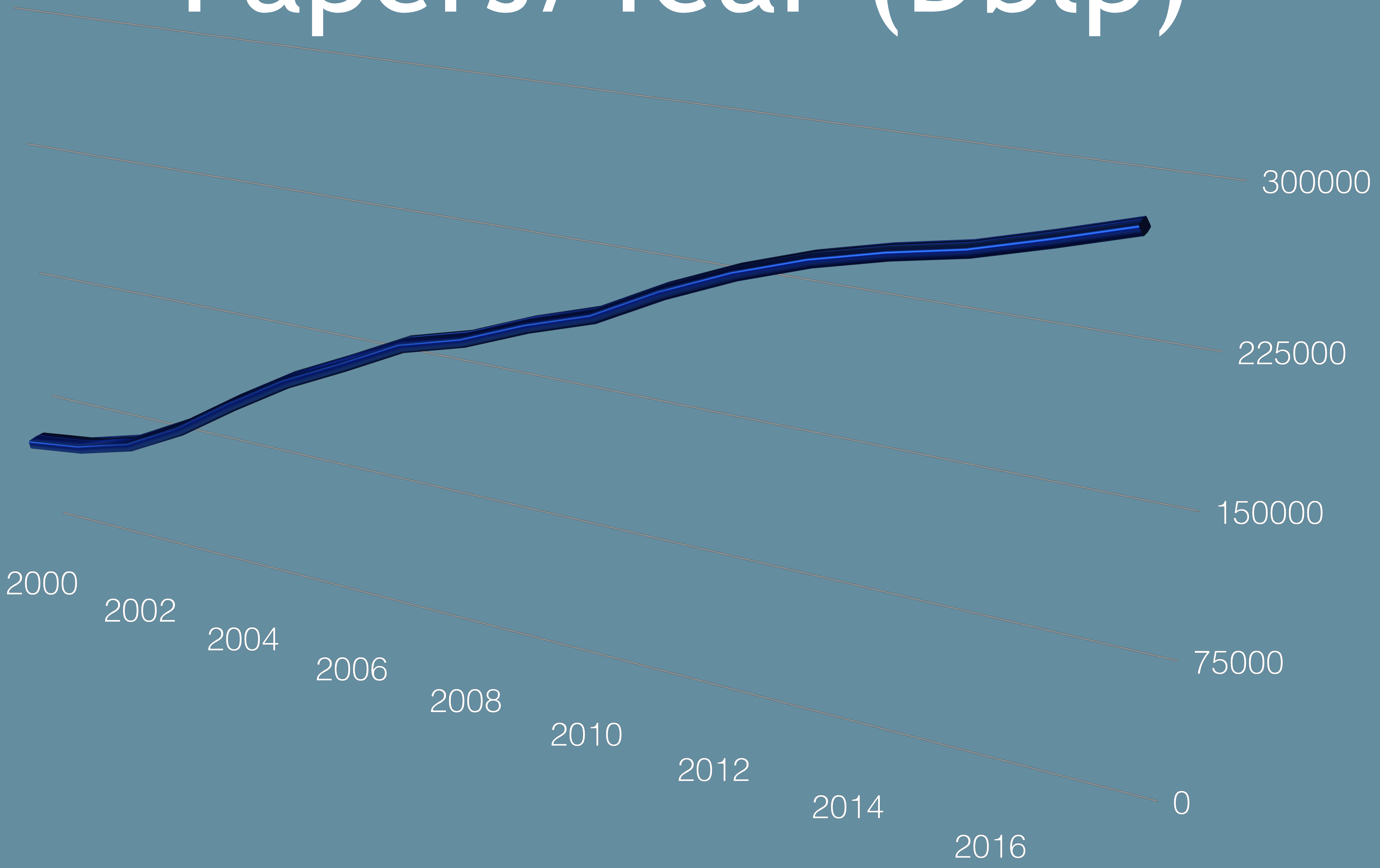
search.org FAQ Privacy policy Contact

ACM Programming Language Design and Implementation, PLDI 2014

- 225 conferences
- 18,000 articles
- 39,000 unique authors
- 64,000 verification emails sent

- 10% of articles are verified
- 6% of articles have shared artifacts

Papers/Year (Dblp)



6th Law of Artifact Sha[illegible]ng

(Inverse Costner's Law)

Even if you built it,
they still wouldn't come.

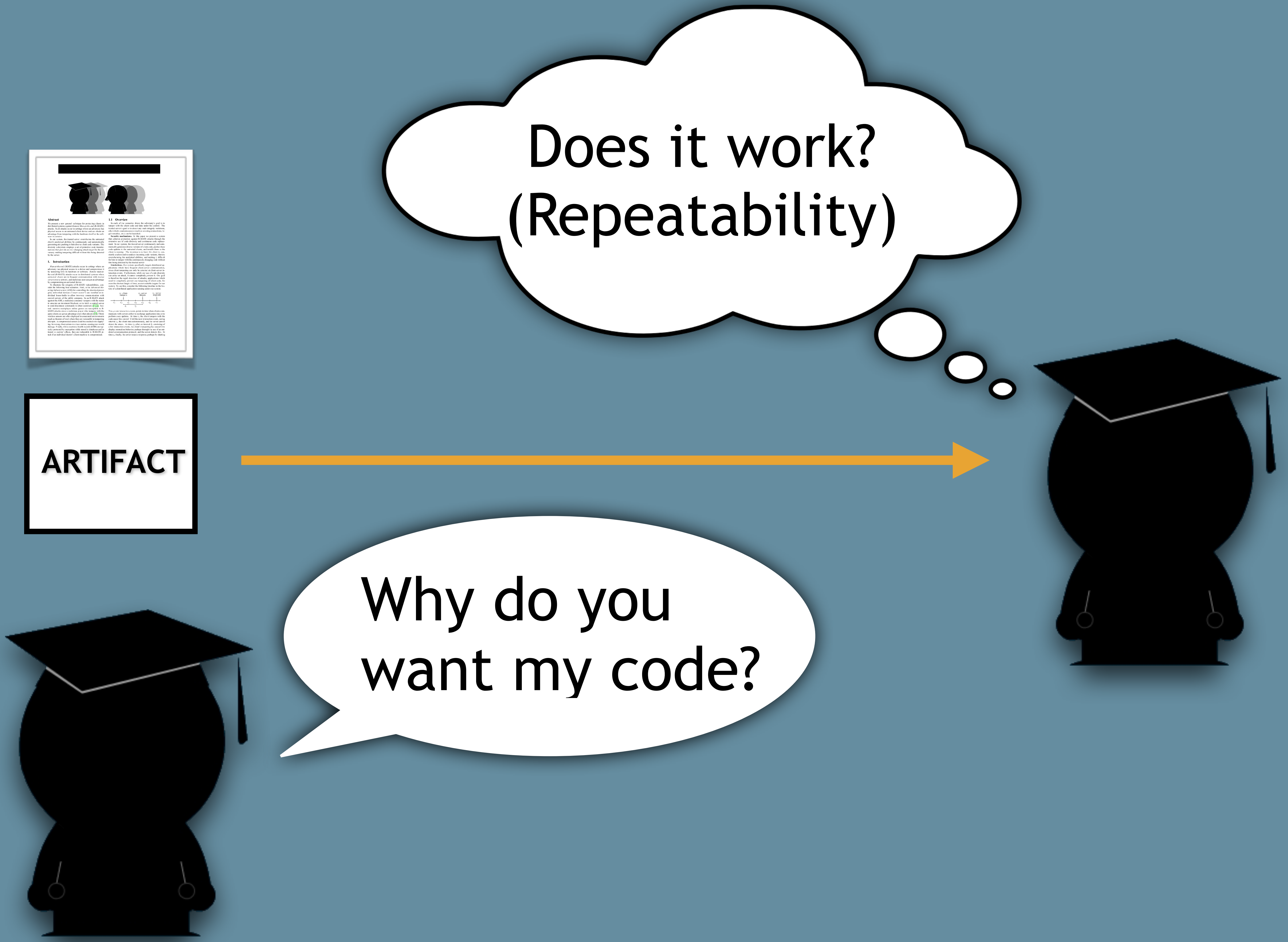
6th Law of Artifact Sha ng

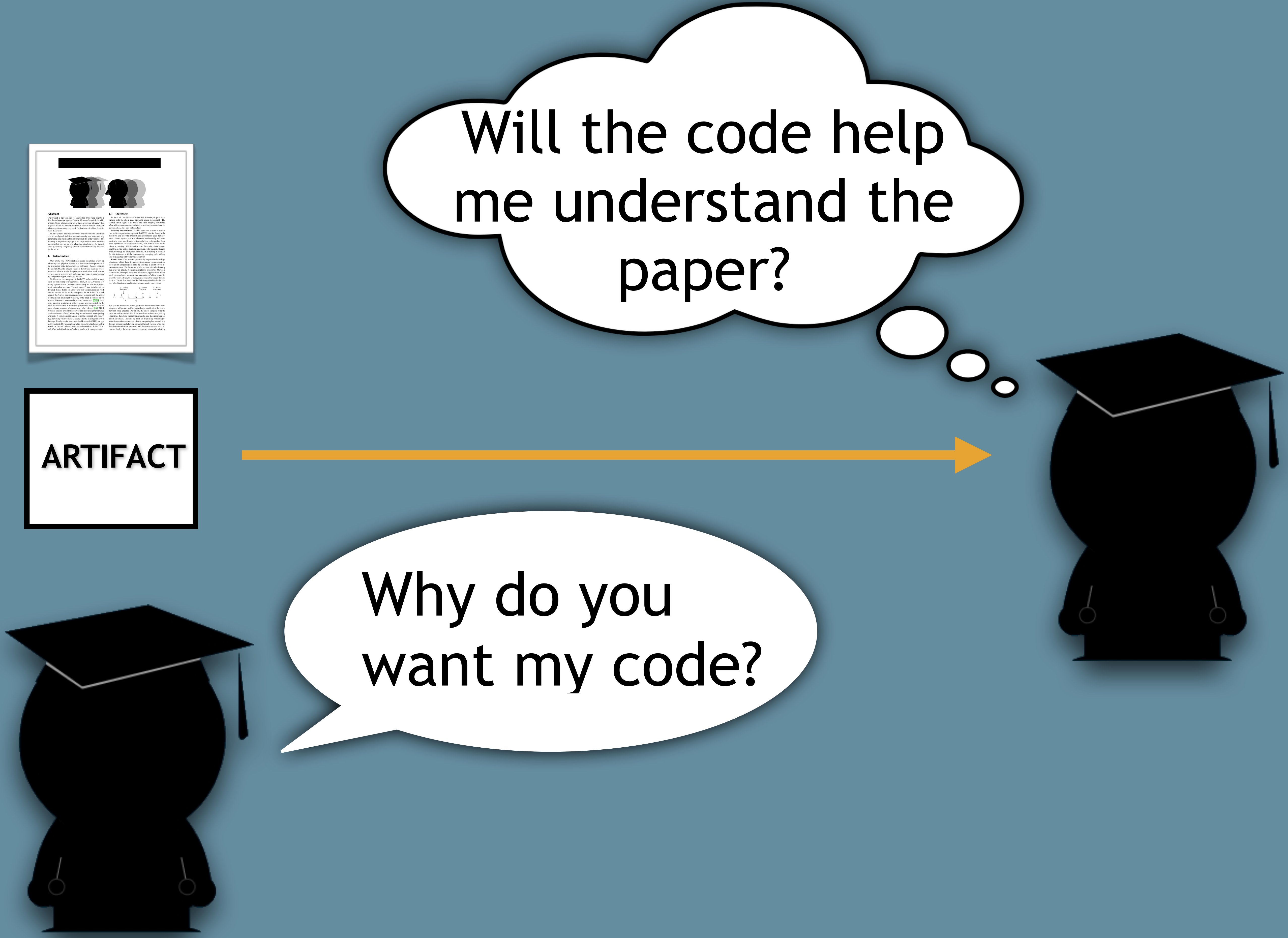
(Inverse Costner's Law)

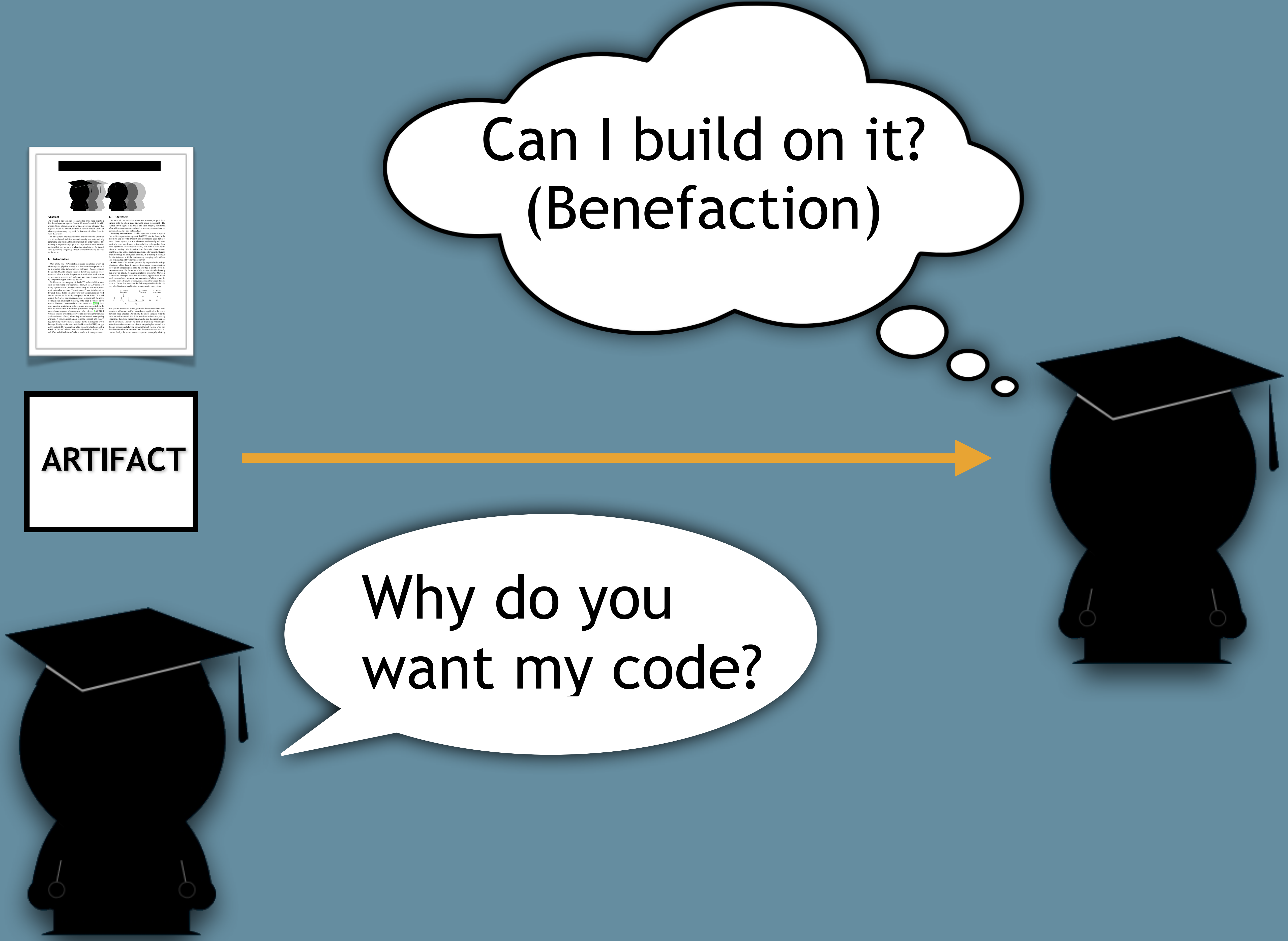
Even if you built it,
they still wouldn't come.

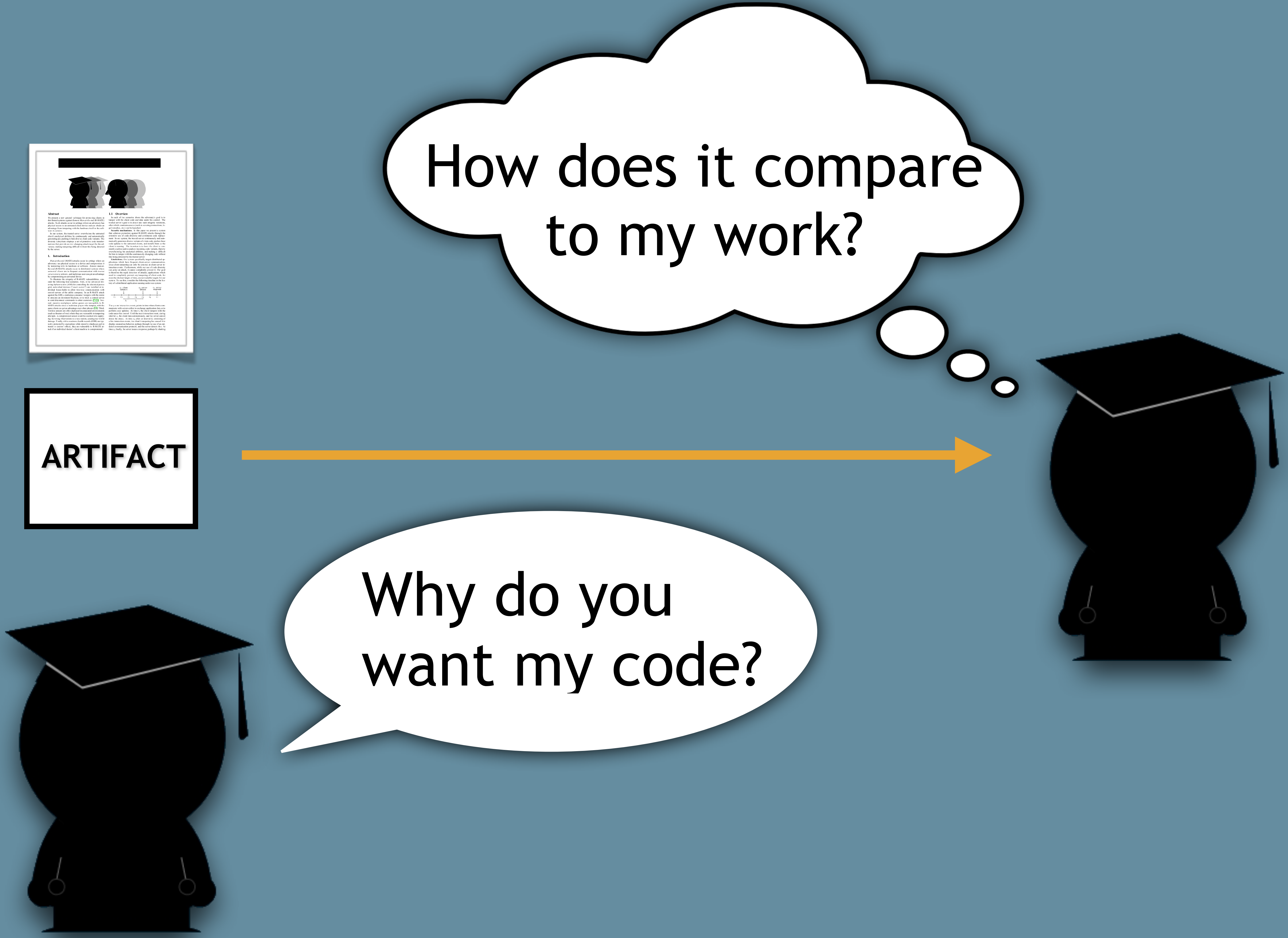
Sharing Proposal — #2 —

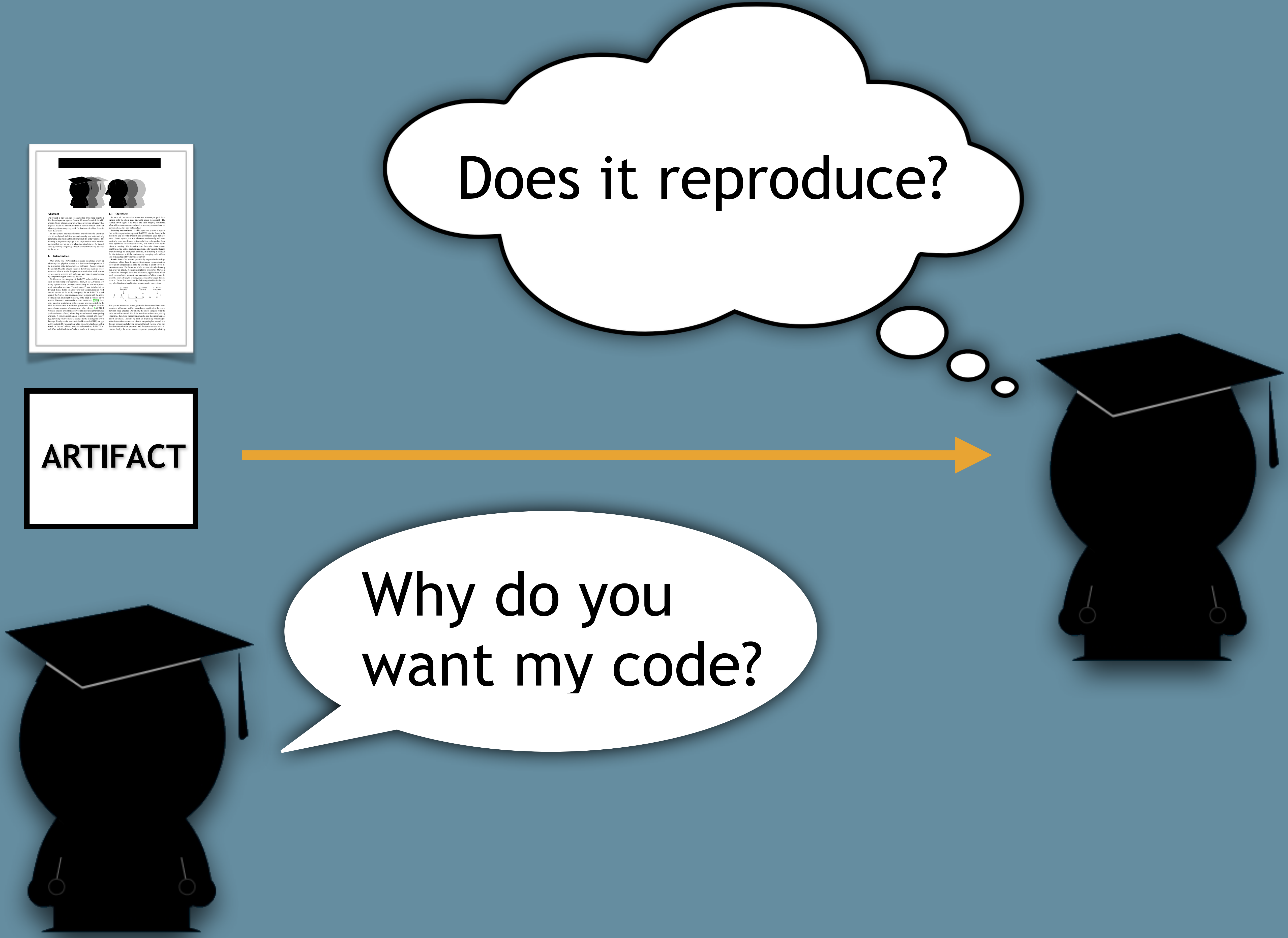
Checklists

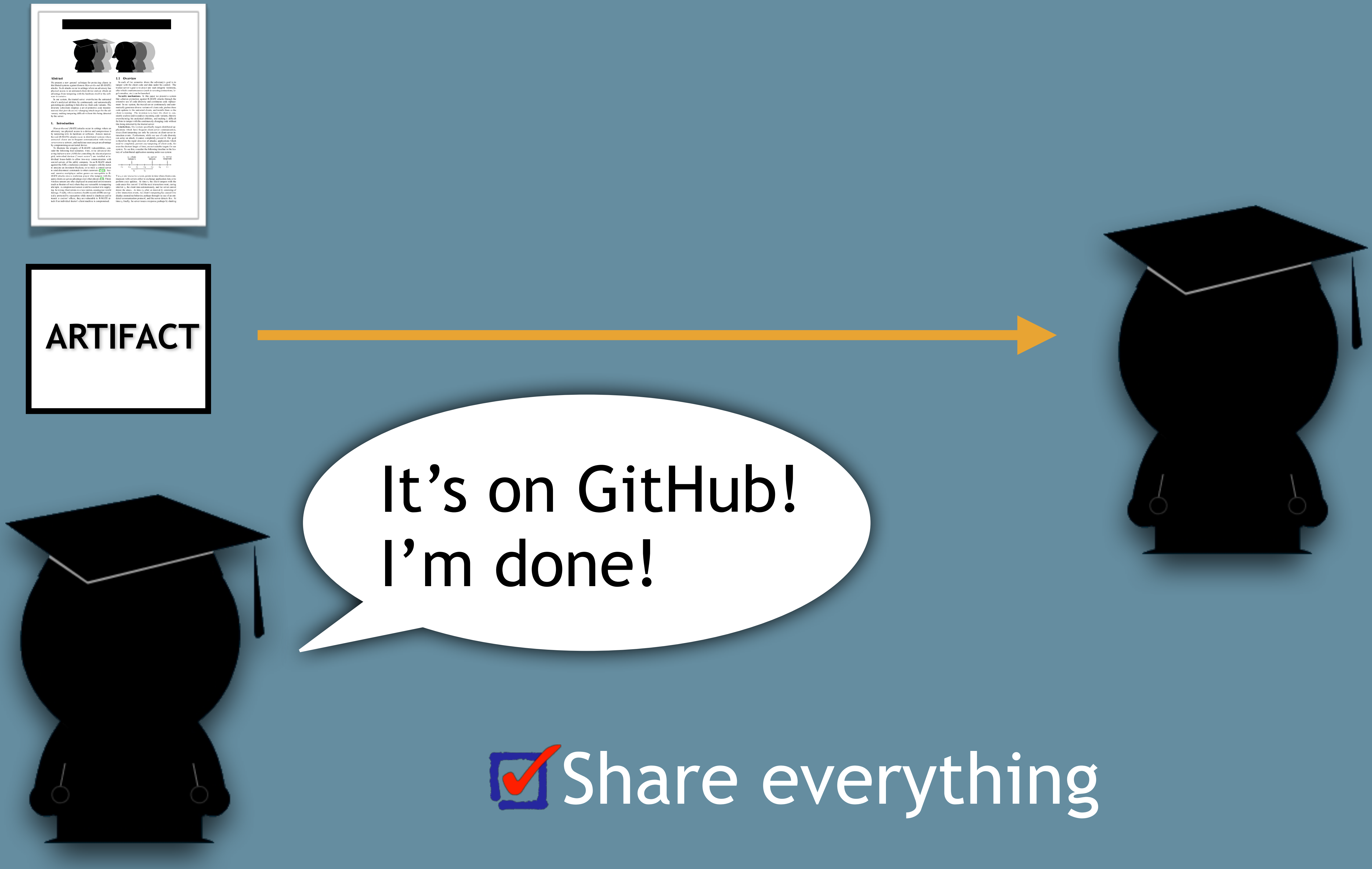

ARTIFACT
Does it work? (Repeatability)
Why do you want my code?


ARTIFACT
Will the code help me understand the paper?
Why do you want my code?

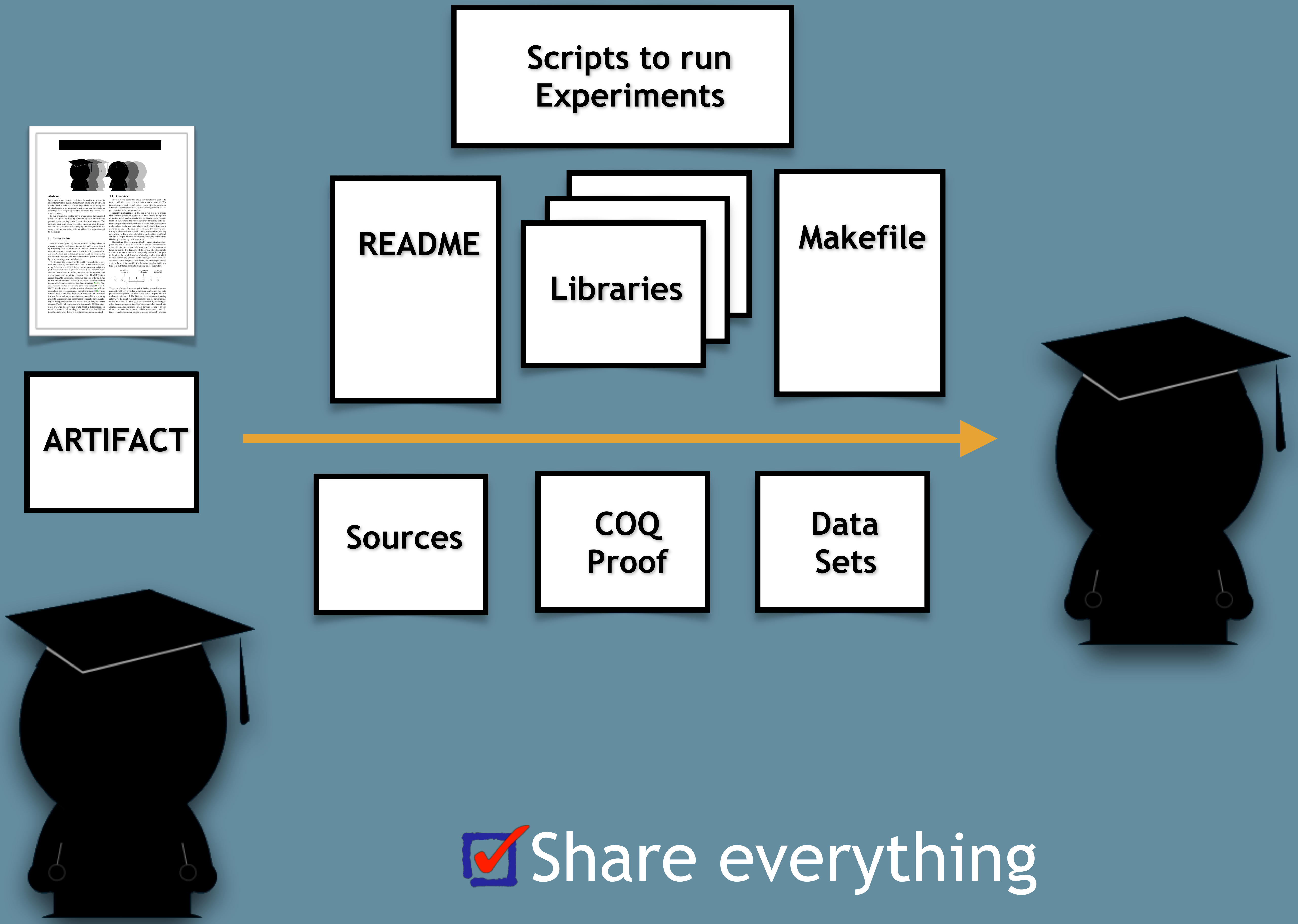

Can I build on it?
(Benefaction)
ARTIFACT
Why do you want my code?


How does it compare to my work?
ARTIFACT
Why do you want my code?

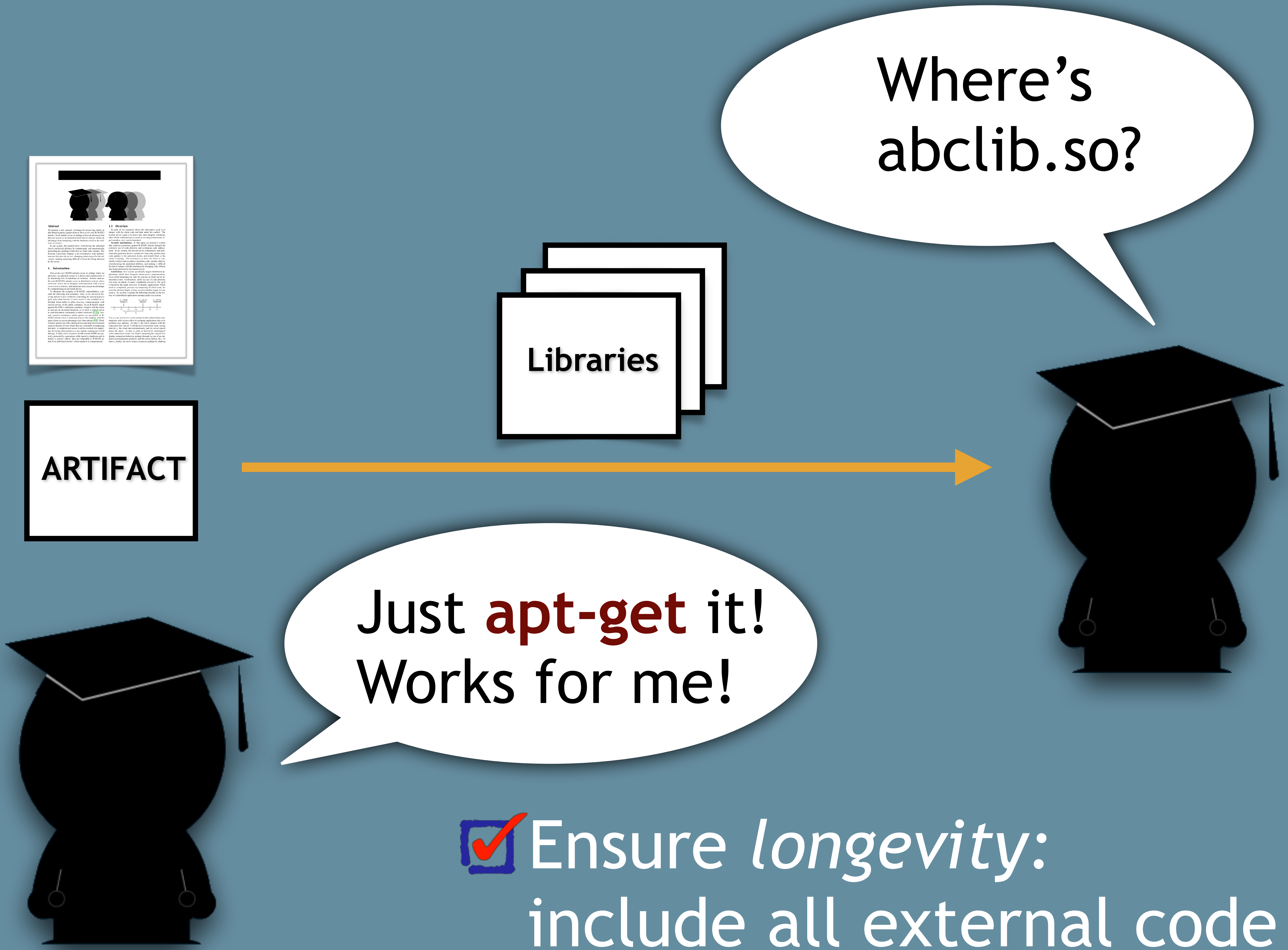

Does it reproduce?
ARTIFACT
Why do you want my code?



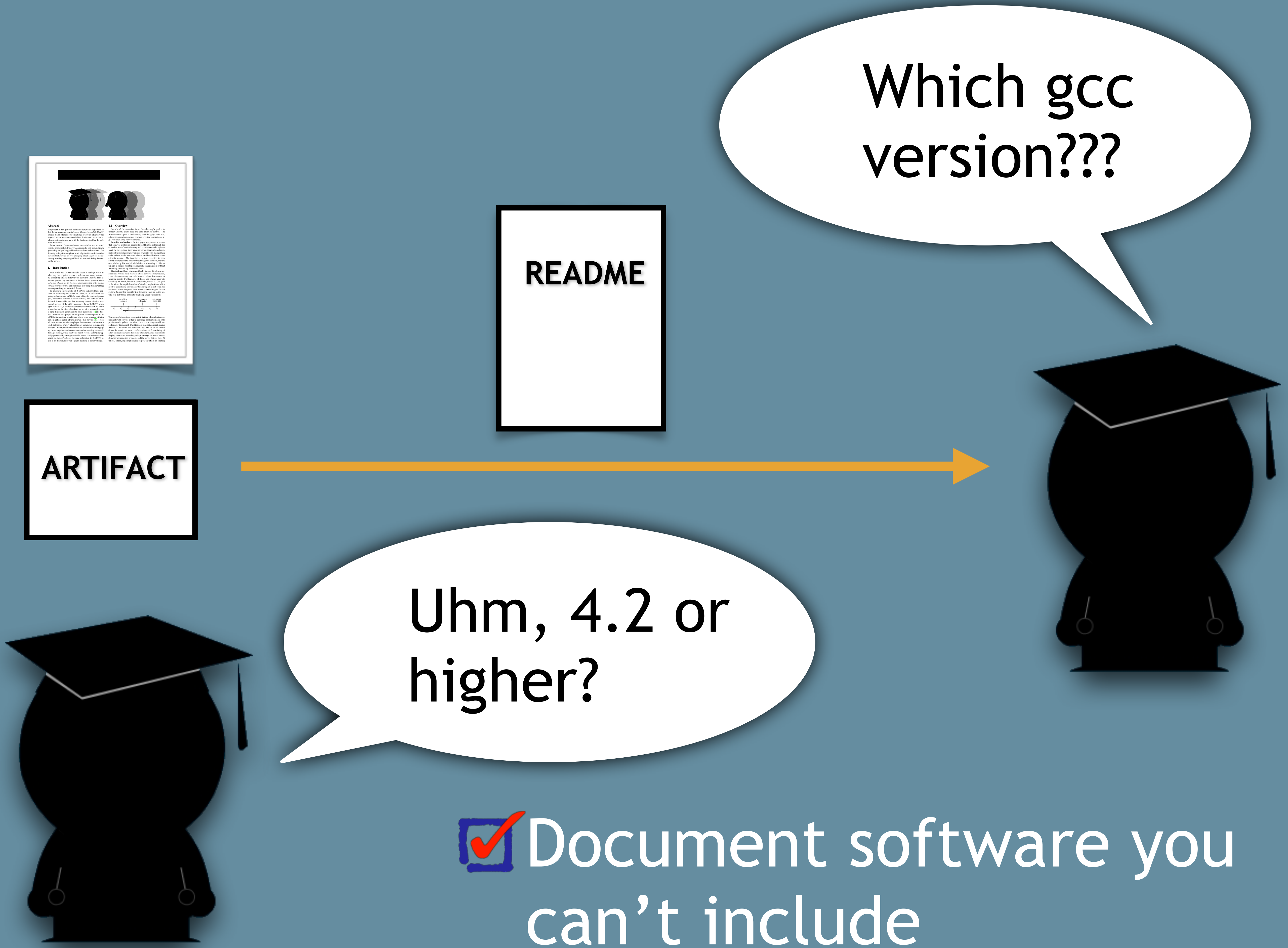
✓ Share everything



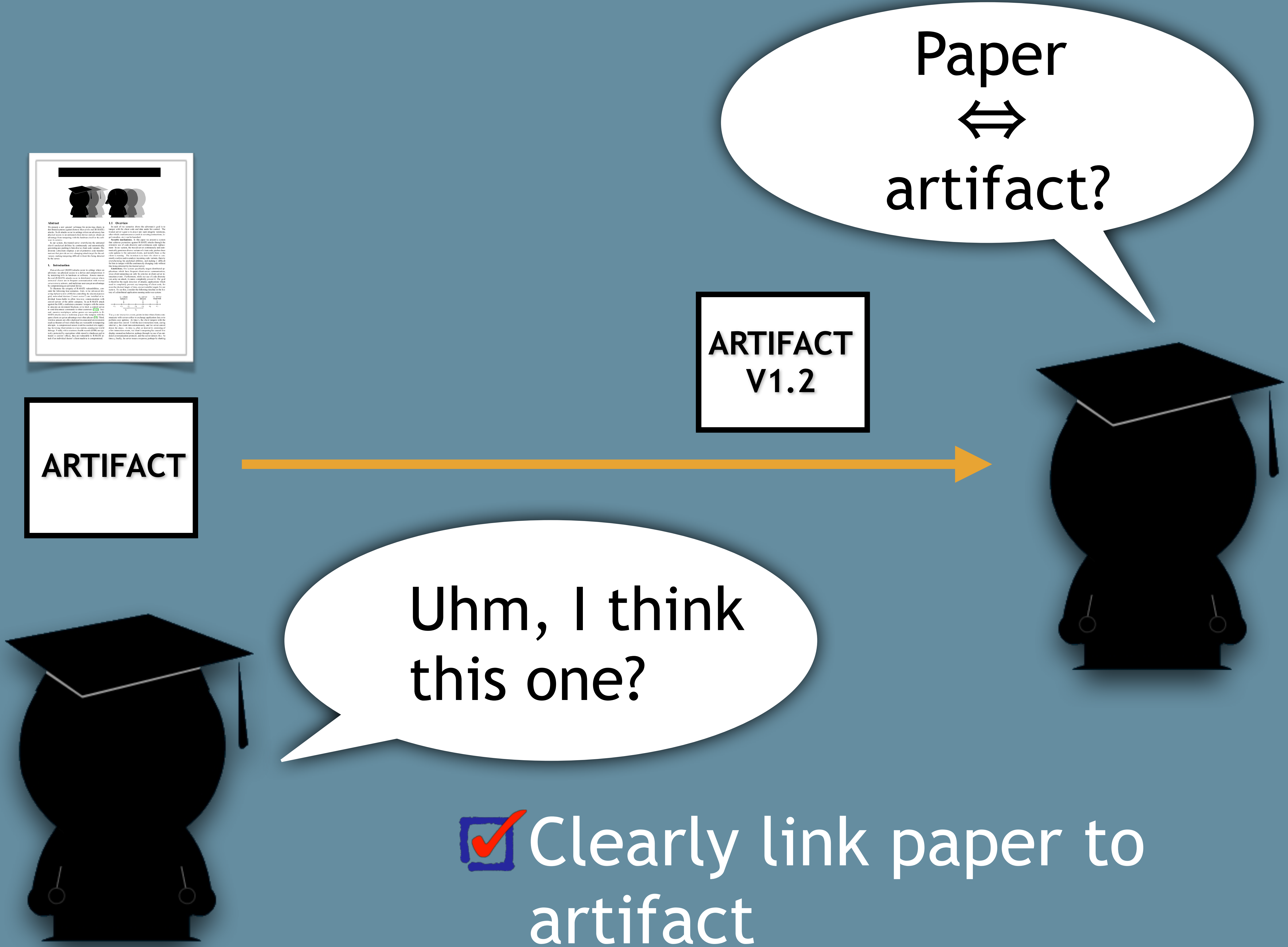
☑ Share everything



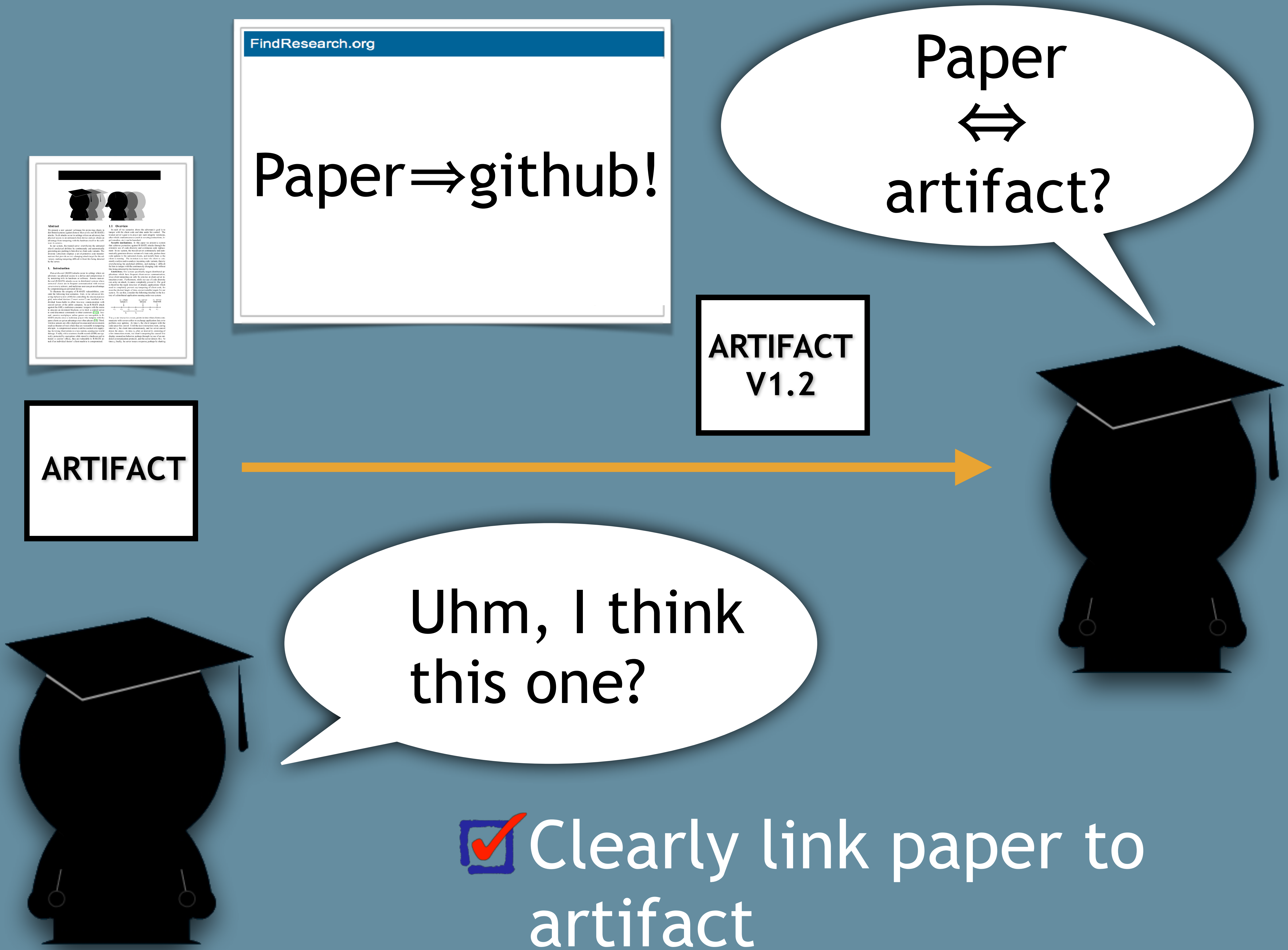
☑ Ensure *longevity*:
include all external code



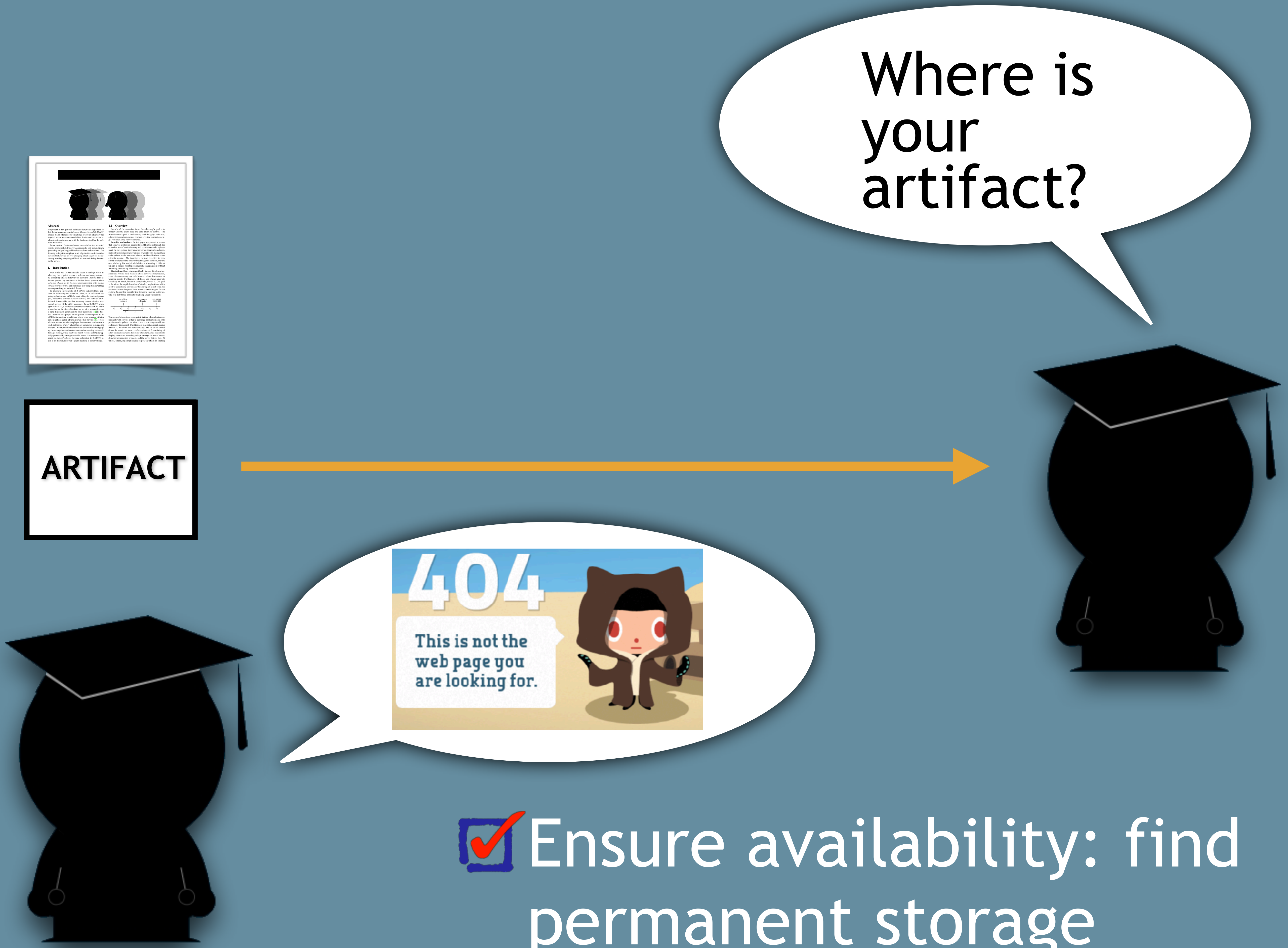
☑ Document software you can't include



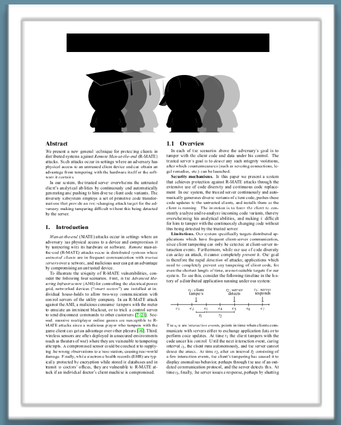
☑ Clearly link paper to artifact

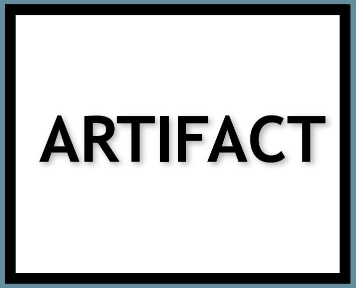


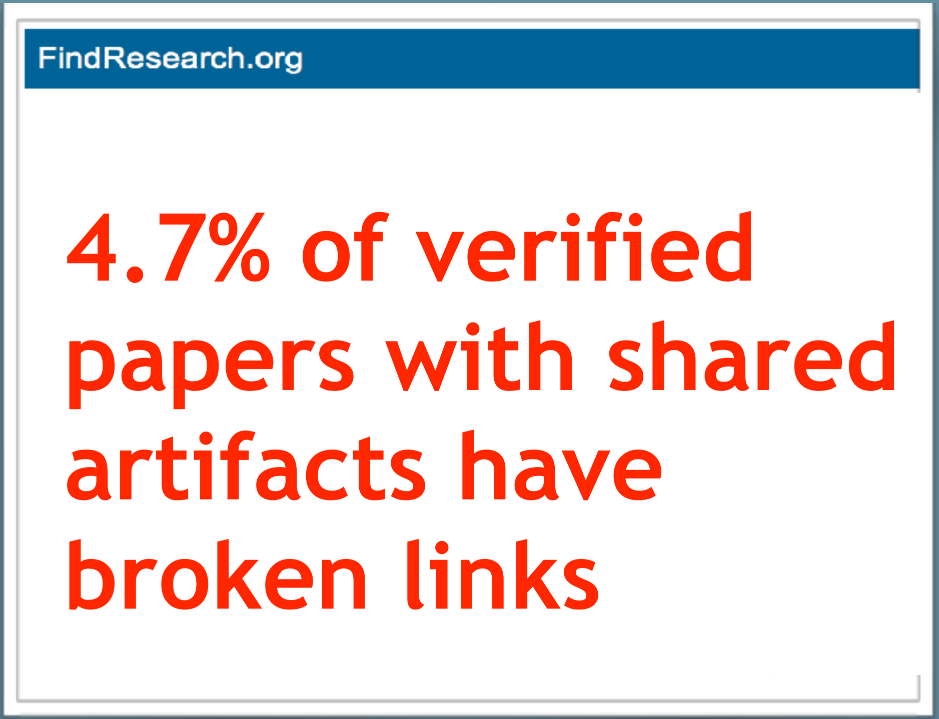
☑ Clearly link paper to artifact



☑ Ensure availability: find permanent storage

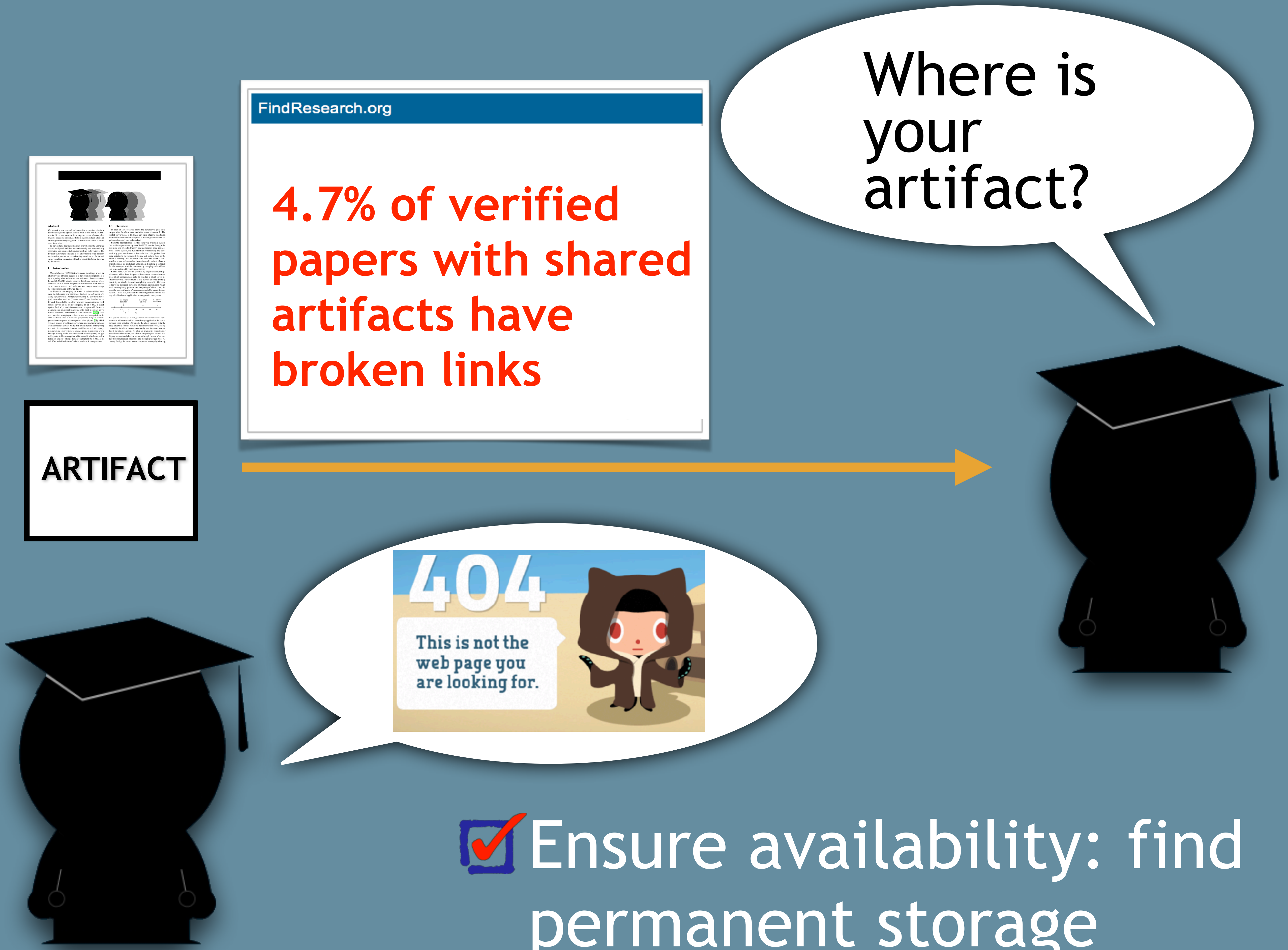




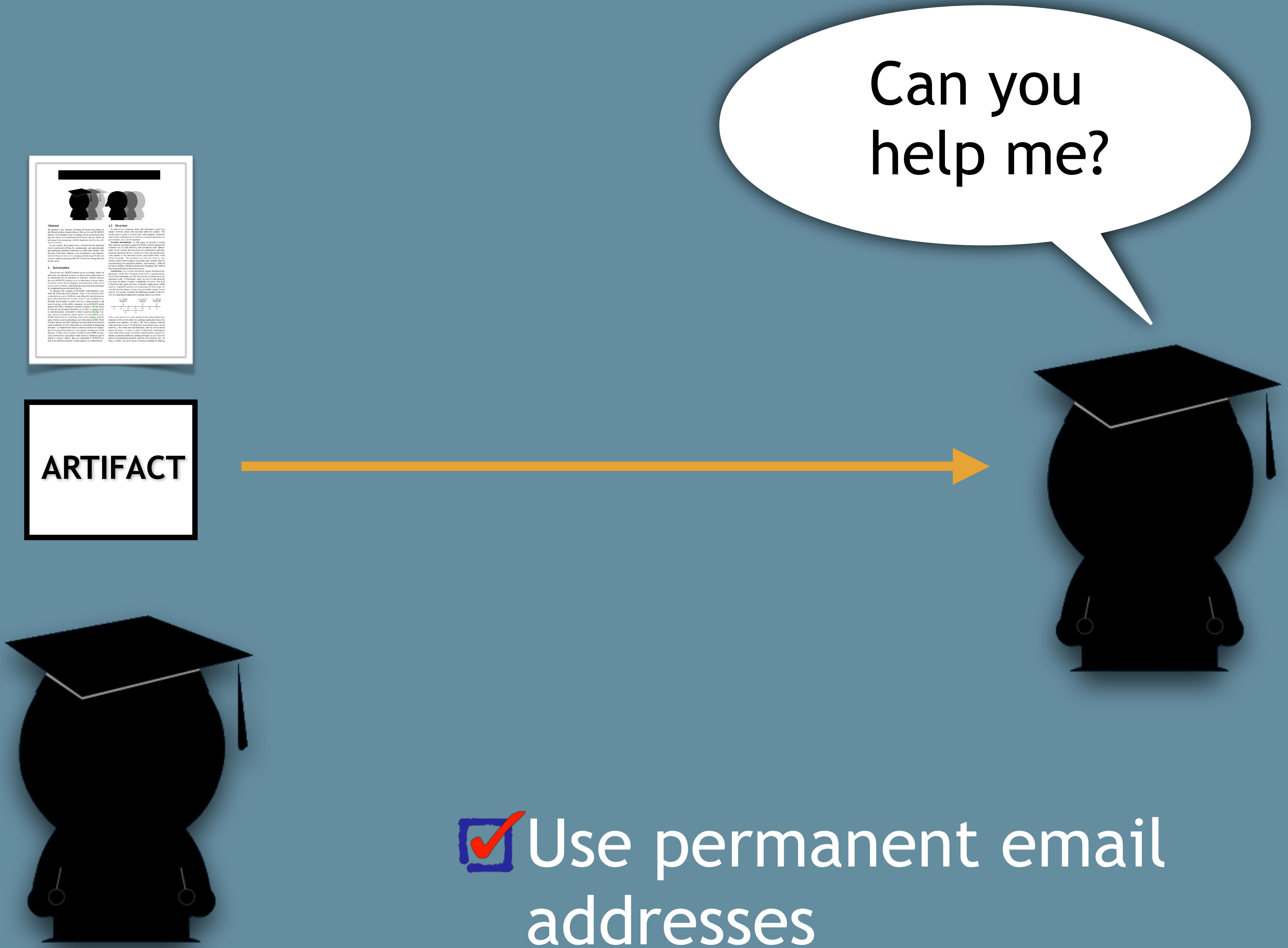


FindResearch.org

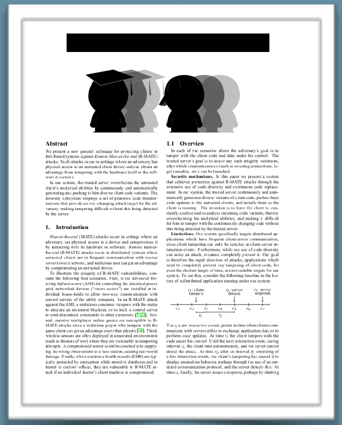
4.7% of verified papers with shared artifacts have broken links



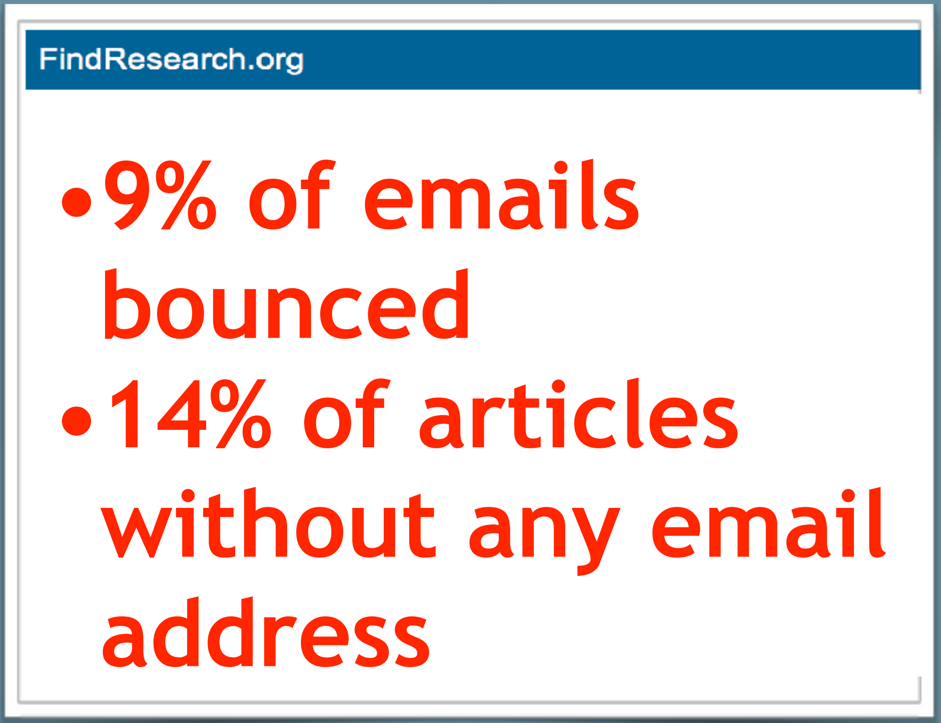
☑ Ensure availability: find permanent storage



☑ Use permanent email addresses



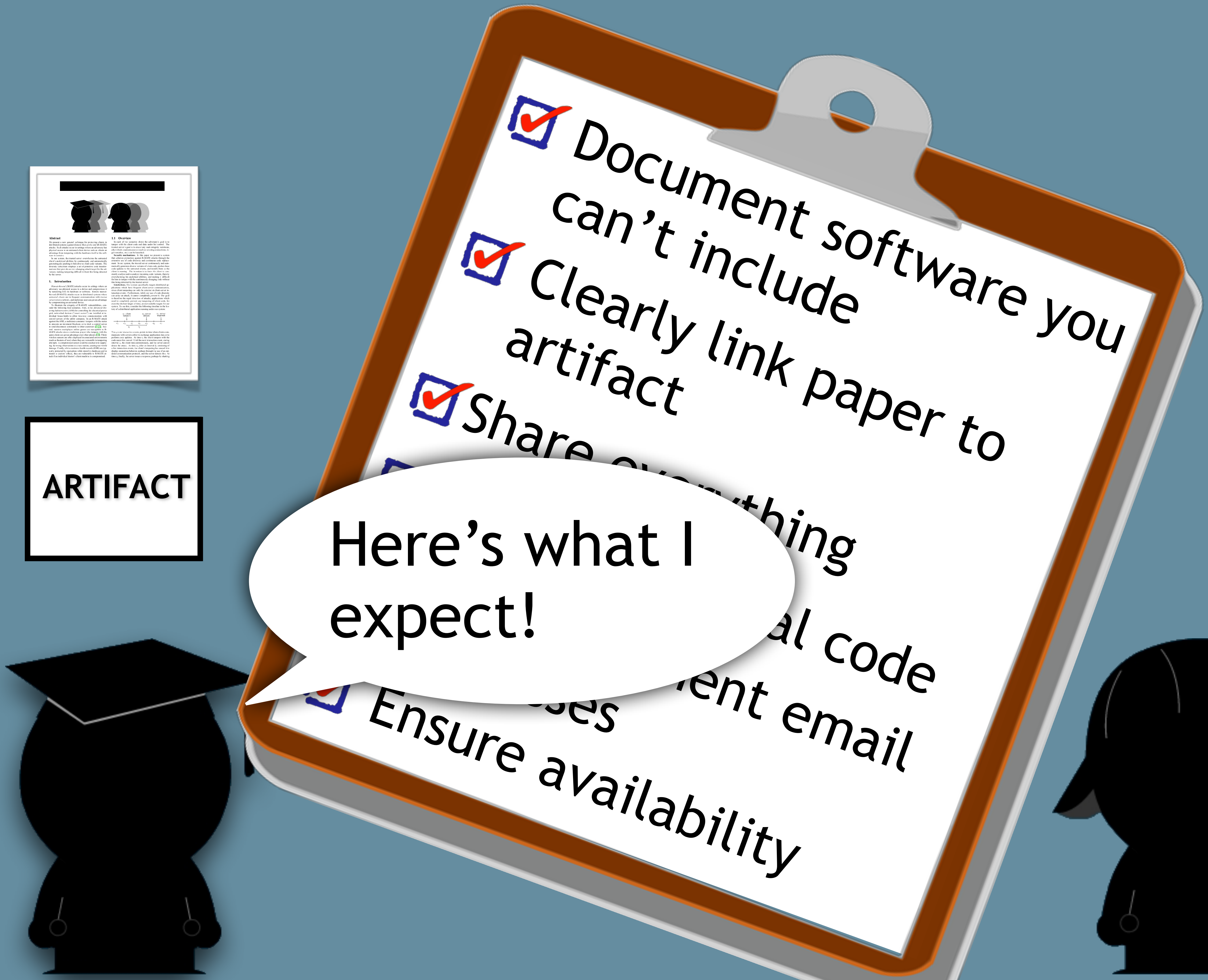
ARTIFACT



FindResearch.org

- 9% of emails bounced
- 14% of articles without any email address

✓ Use permanent email addresses

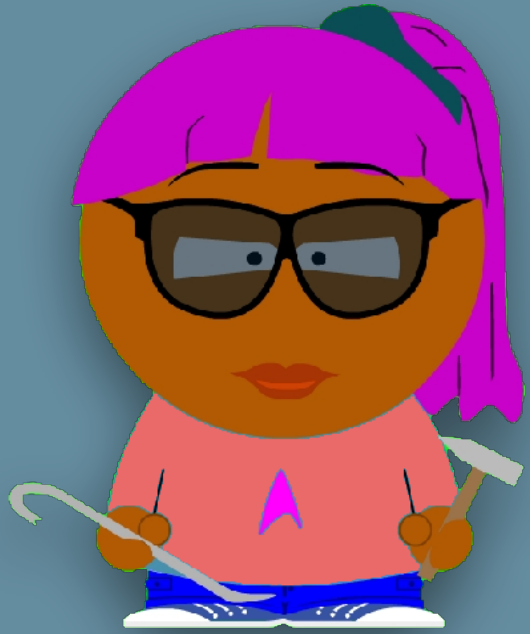

ARTIFACT
Document software you can't include
Clearly link paper to artifact
Share everything
al code
ent email
ses
Ensure availability
Here's what I expect!

5th Law of Artifact Sharing

To ensure repeatability of your results by others, you must

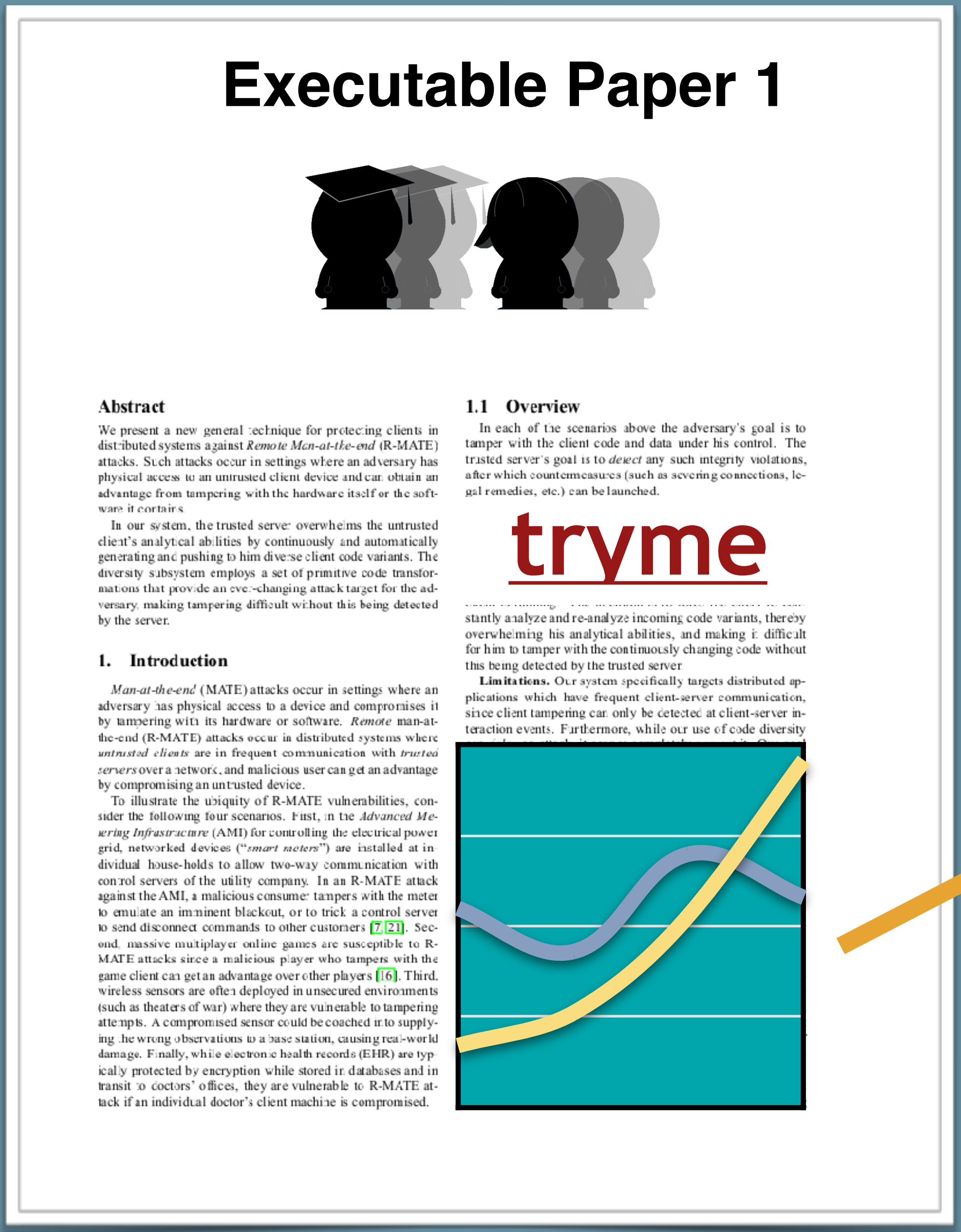
1. share everything
2. assume nothing
3. remain reachable

Sharing Proposal — #3 —



Tool Support

is.ieis.tue.nl/staff/pvgorp/share



Executable Paper 1

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.

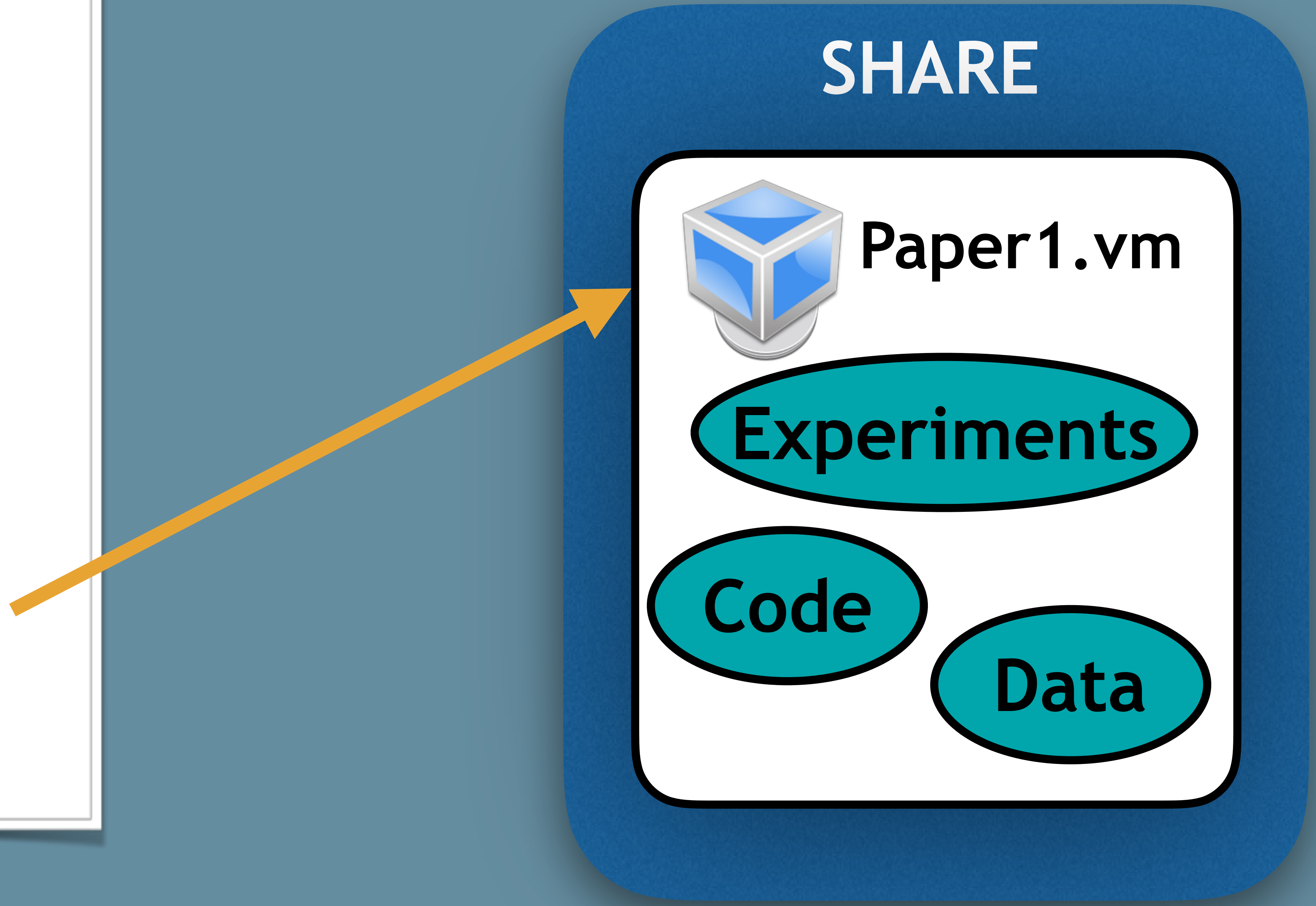
1.1 Overview

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.

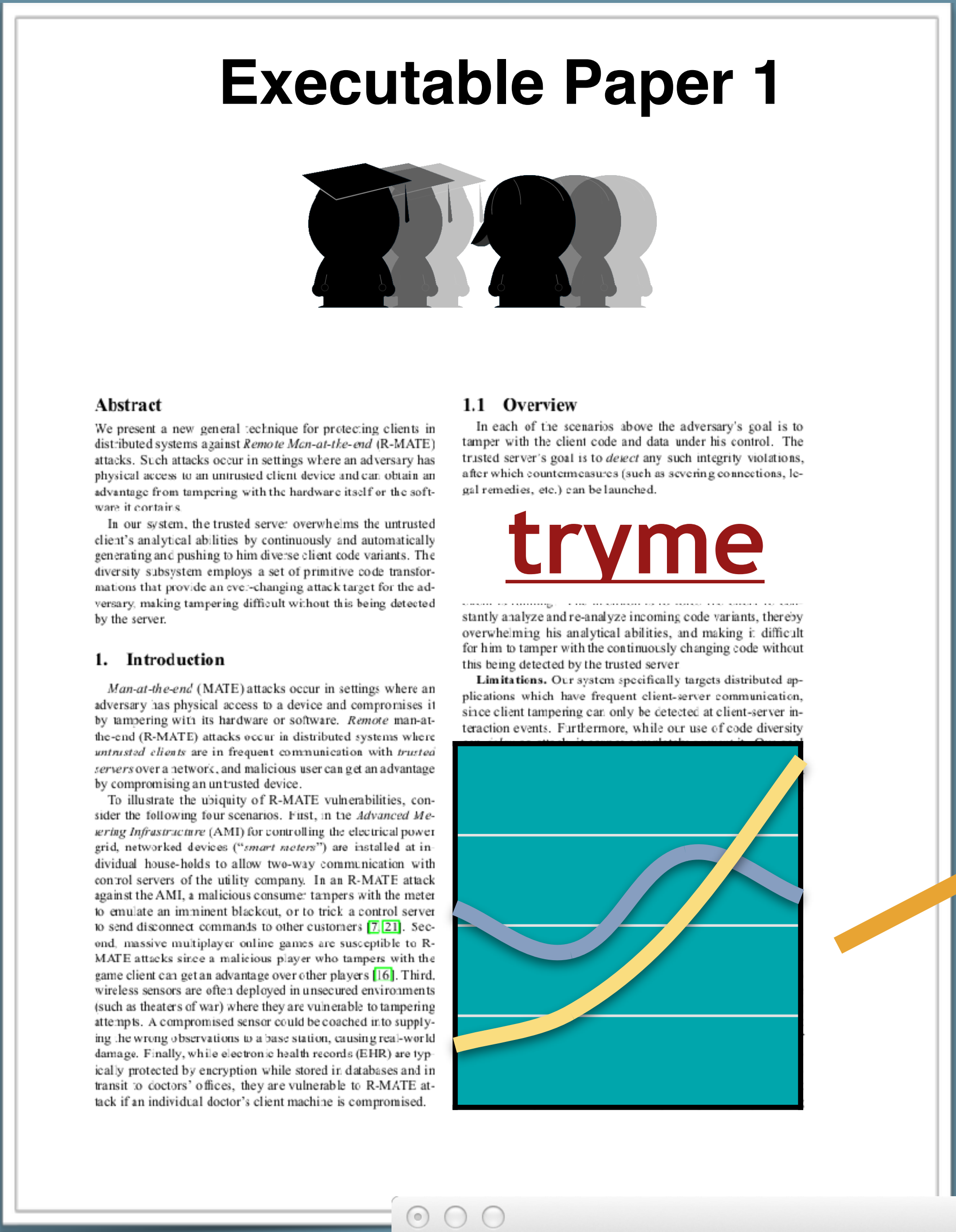
tryme

stantly 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



is.ieis.tue.nl/staff/pvgorp/share



Executable Paper 1

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.

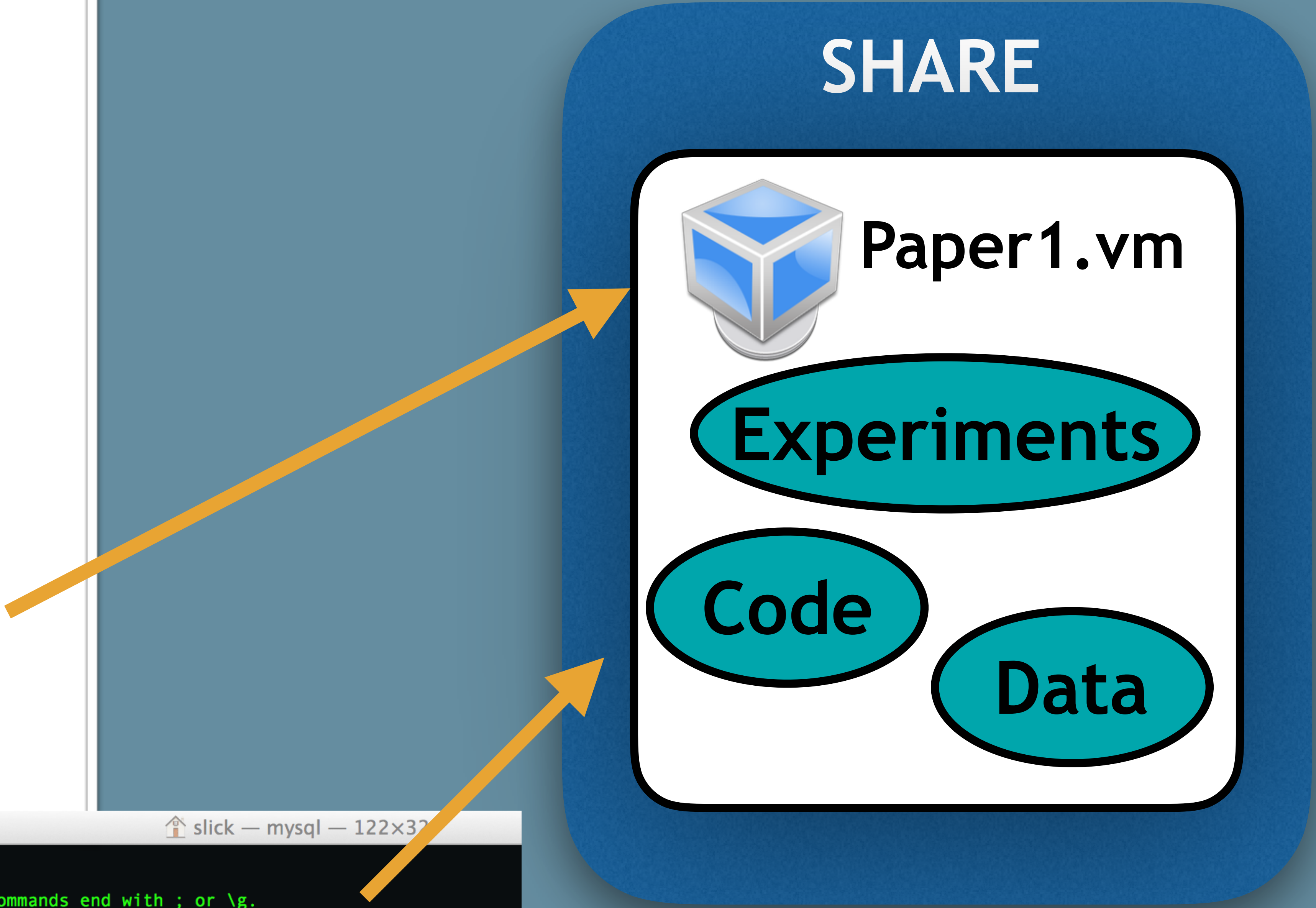
1.1 Overview

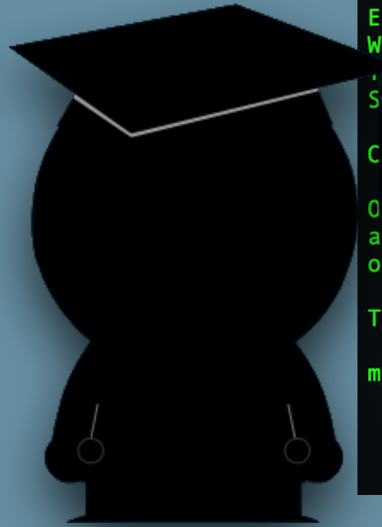
In each of the scenarios above the adversary's goal is to tamper with the client code and data under his control. The trusted server's goal is to *detect* any such integrity violations, after which countermeasures (such as severing connections, legal remedies, etc.) can be launched.

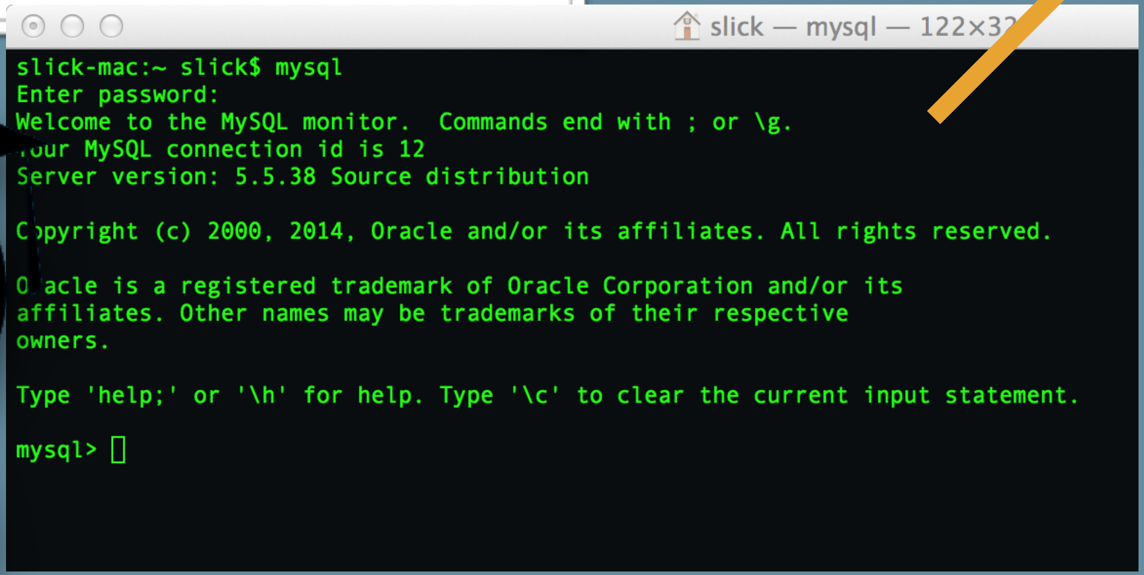
tryme

stantly 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

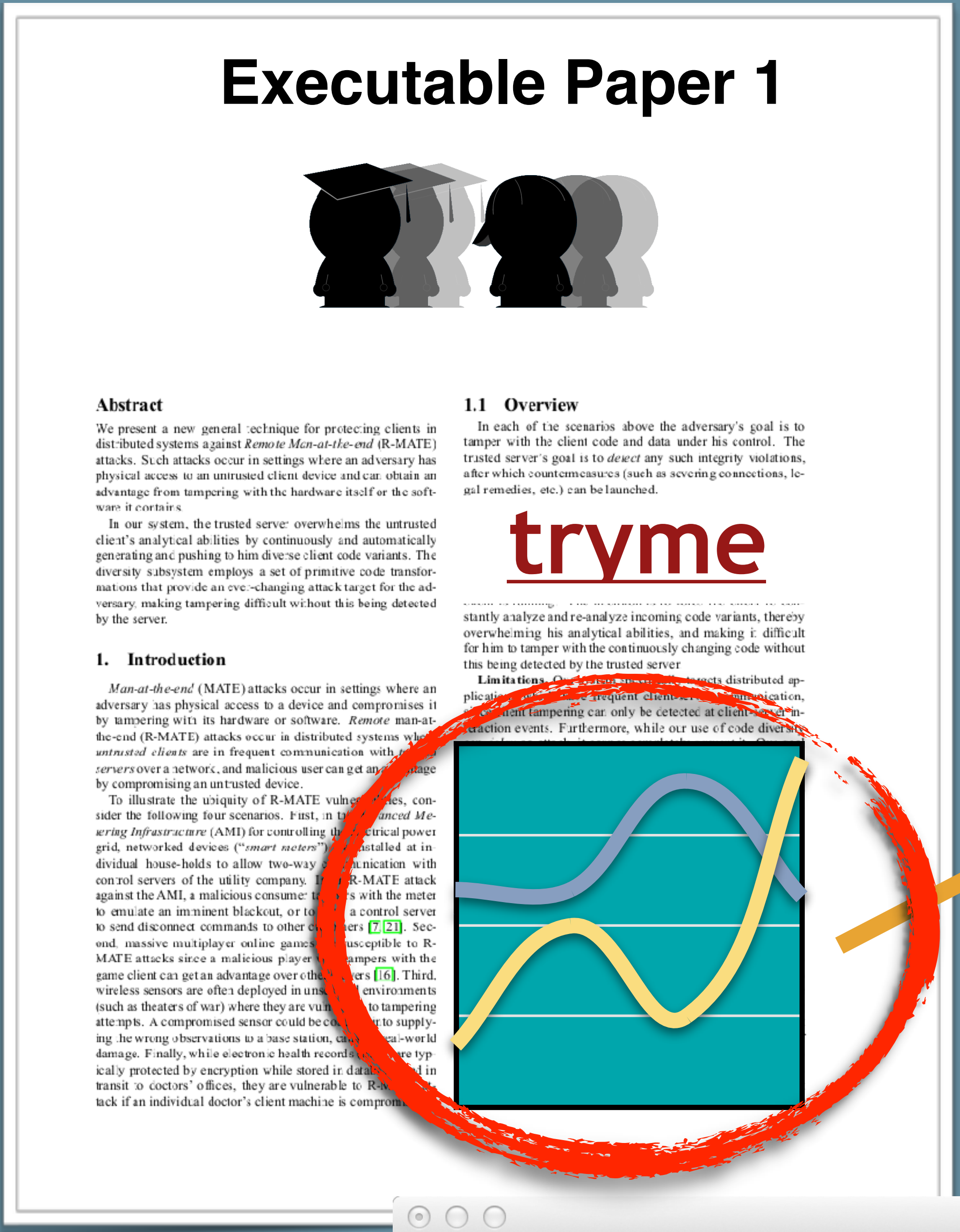






```
slick-mac:~ slick$ mysql
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.5.38 Source distribution

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```

is.ieis.tue.nl/staff/pvgorp/share



Executable Paper 1

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 whe[illegible] *untrusted clients* are in frequent communication with [illegible] *servers* over a network, and malicious user can get an [illegible]age by compromising an untrusted device.

To illustrate the ubiquity of R-MATE vulne[illegible]es, consider the following four scenarios. First, in t[illegible]*anced Metering Infrastructure* (AMI) for controlling th[illegible]ctrical power grid, networked devices ("*smart meters*") [illegible]stalled at individual house-holds to allow two-way c[illegible]nication with control servers of the utility company. I[illegible] R-MATE attack against the AMI, a malicious consumer t[illegible]s with the meter to emulate an imminent blackout, or to [illegible] a control server to send disconnect commands to other c[illegible]ers [7, 21]. Second, massive multiplayer online games [illegible] susceptible to R-MATE attacks since a malicious player [illegible]ampers with the game client can get an advantage over othe[illegible]ers [16]. Third, wireless sensors are often deployed in uns[illegible] environments (such as theaters of war) where they are vul[illegible] to tampering attempts. A compromised sensor could be co[illegible] to supplying the wrong observations to a base station, ca[illegible]eal-world damage. Finally, while electronic health records [illegible]re typically protected by encryption while stored in data[illegible]d in transit to doctors' offices, they are vulnerable to R-[illegible]tack if an individual doctor's client machine is comprom[illegible]

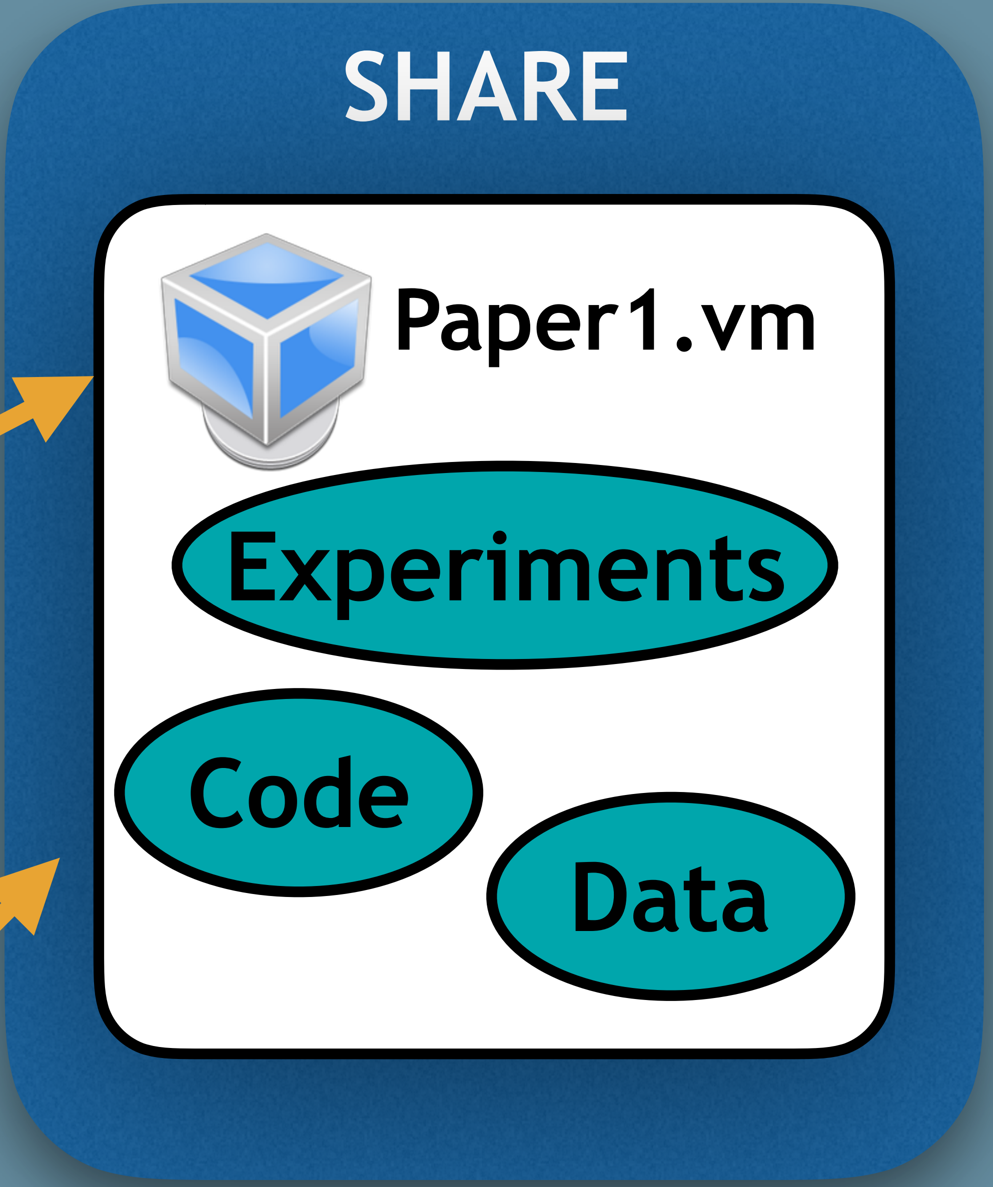
1.1 Overview

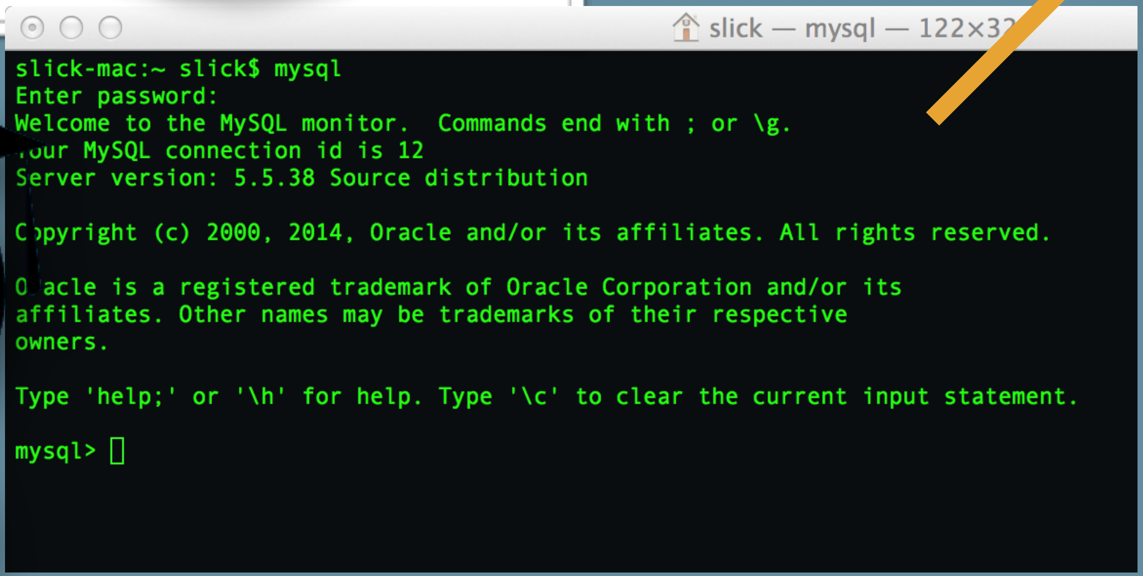
In each of the scenarios above the adversary's goal is to tamper with the client code and data under his control. The trusted server's goal is to *detect* any such integrity violations, after which countermeasures (such as severing connections, legal remedies, etc.) can be launched.

tryme

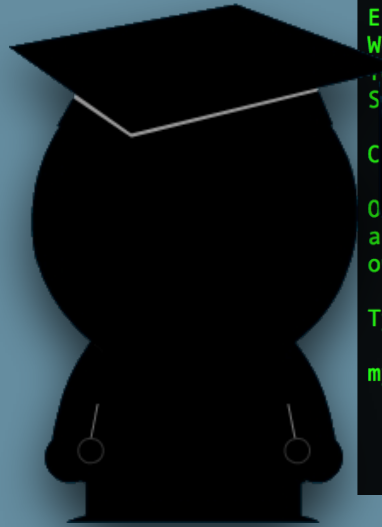
[illegible] stantly analyze and re-analyze incoming code variants, thereby overwhelming his analytical abilities, and making it difficult for him to tamper with the continuously changing code without this being detected by the trusted server.

Limitations. [illegible] distributed applicati[illegible] frequent client-se[illegible]nication, si[illegible]ent tampering can only be detected at client-server in[illegible]eraction events. Furthermore, while our use of code divers[illegible]





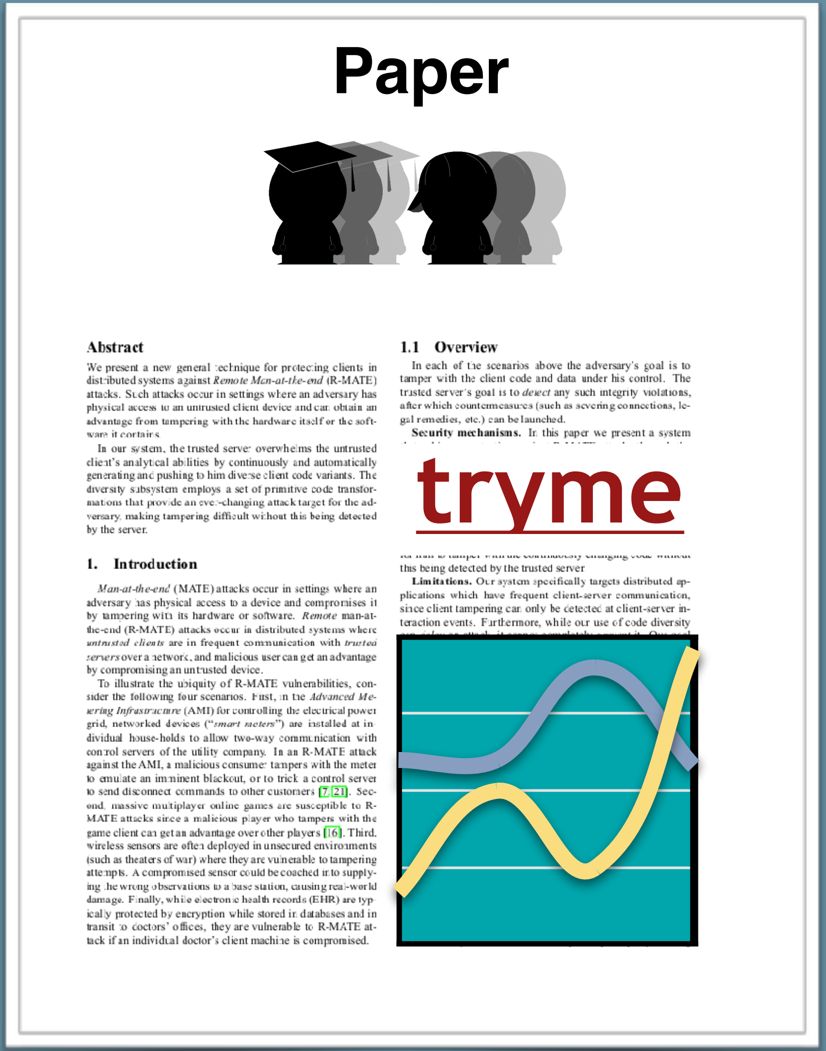
```
slick-mac:~ slick$ mysql
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 5.5.38 Source distribution

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```



VisTrails

www.vistrails.org

Workflow v1.0


Paper

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.

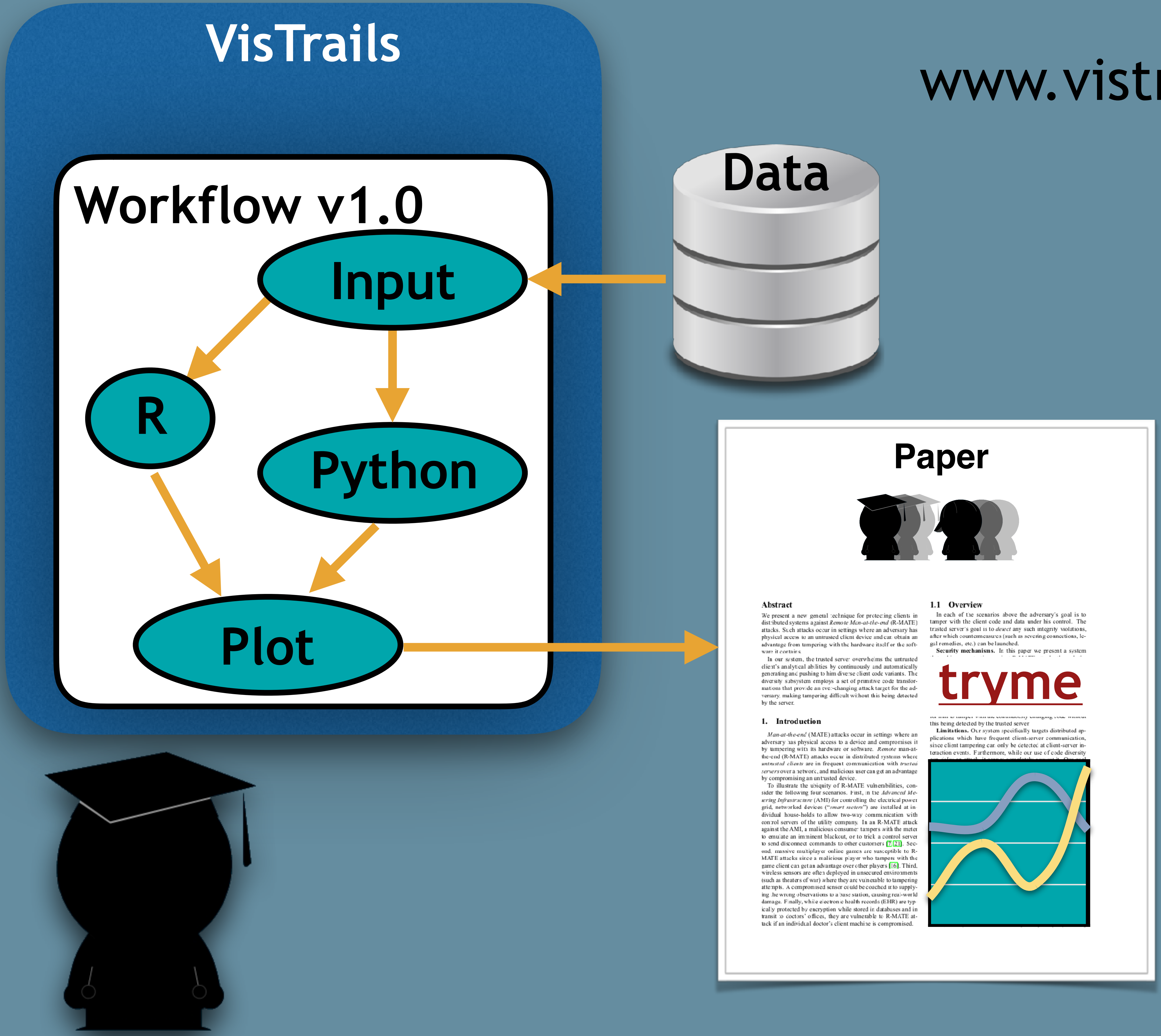
Security mechanisms. In this paper we present a system

tryme

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

www.vistrails.org



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.

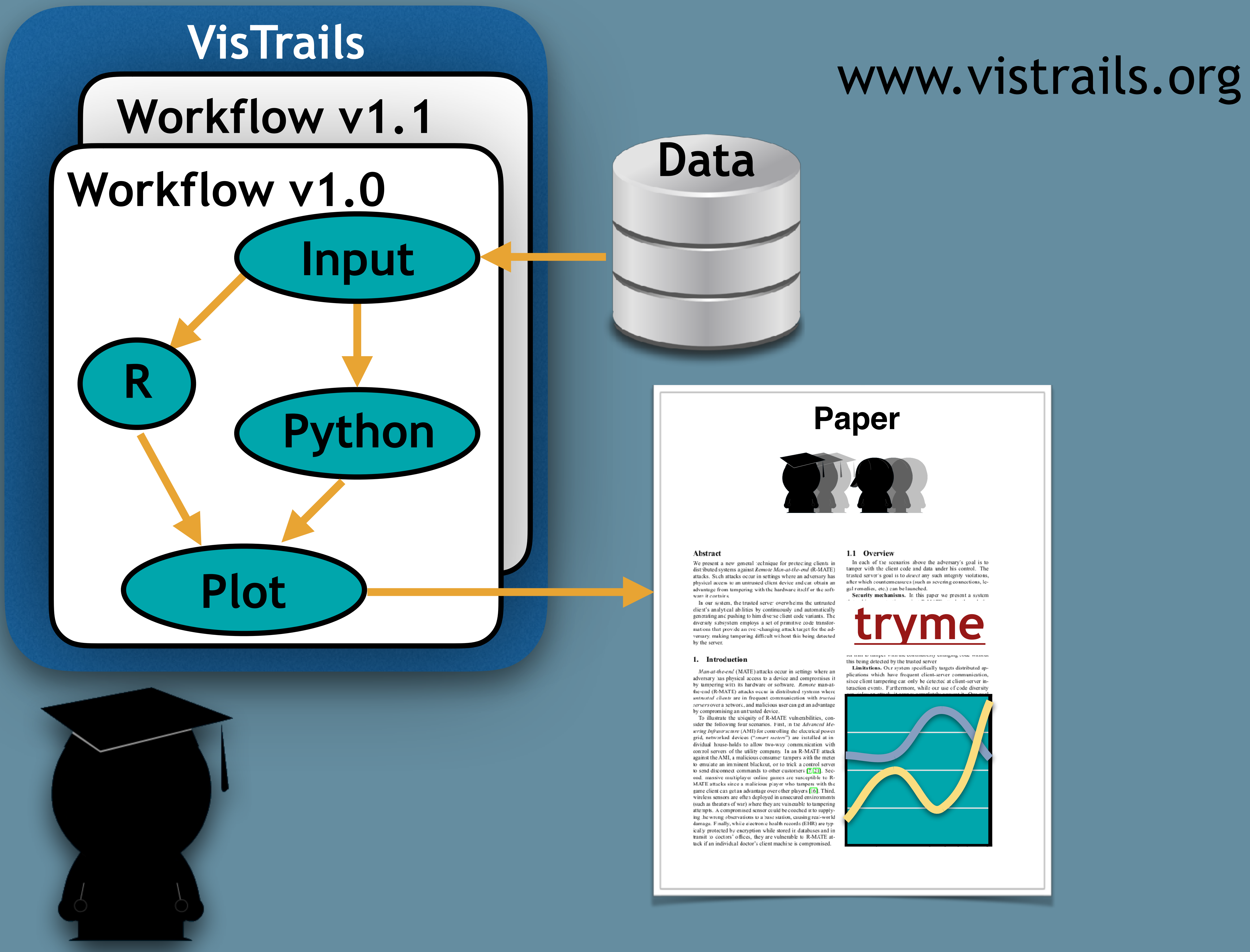
1.1 Overview

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

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



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

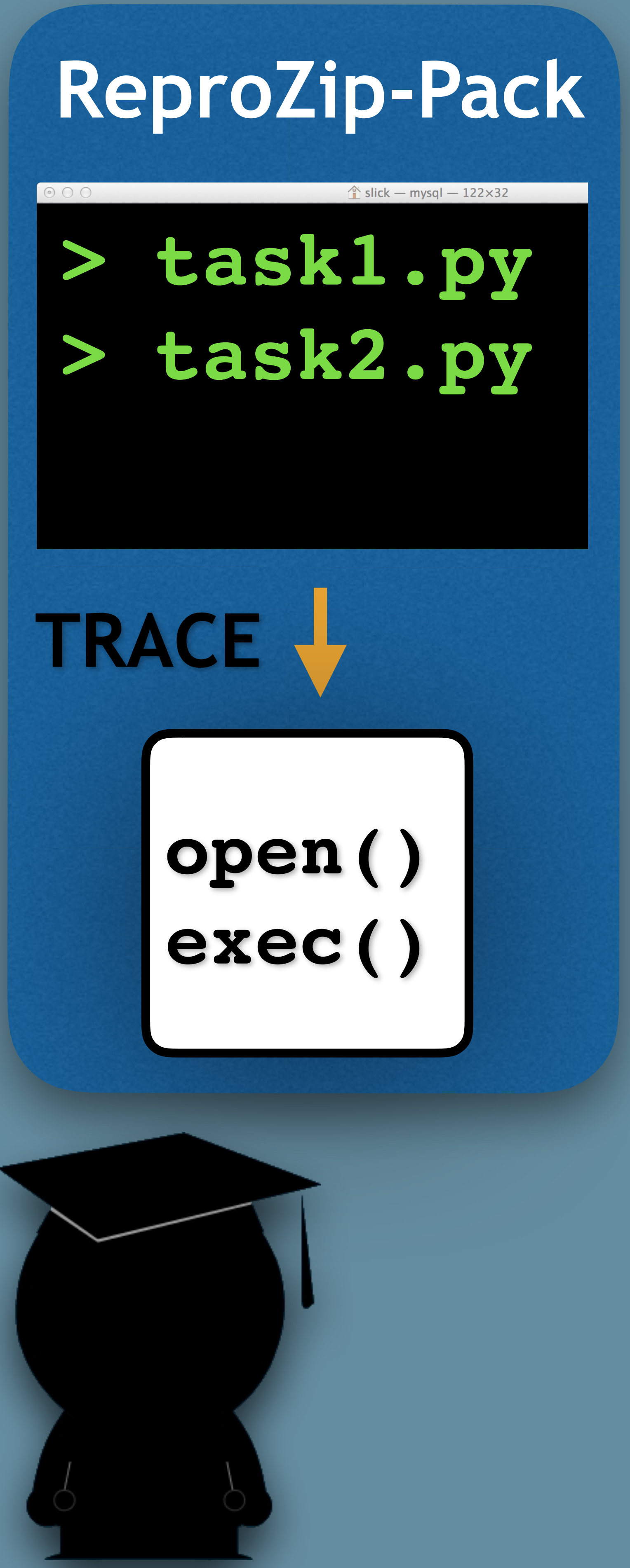
In each of the scenarios above the adversary's goal is to tamper with the client code and data under his control. The trusted server's goal is to *detect* any such integrity violations, after which countermeasures (such as severing connections, legal remedies, etc.) can be launched.

Security mechanisms. In this paper we present a system

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

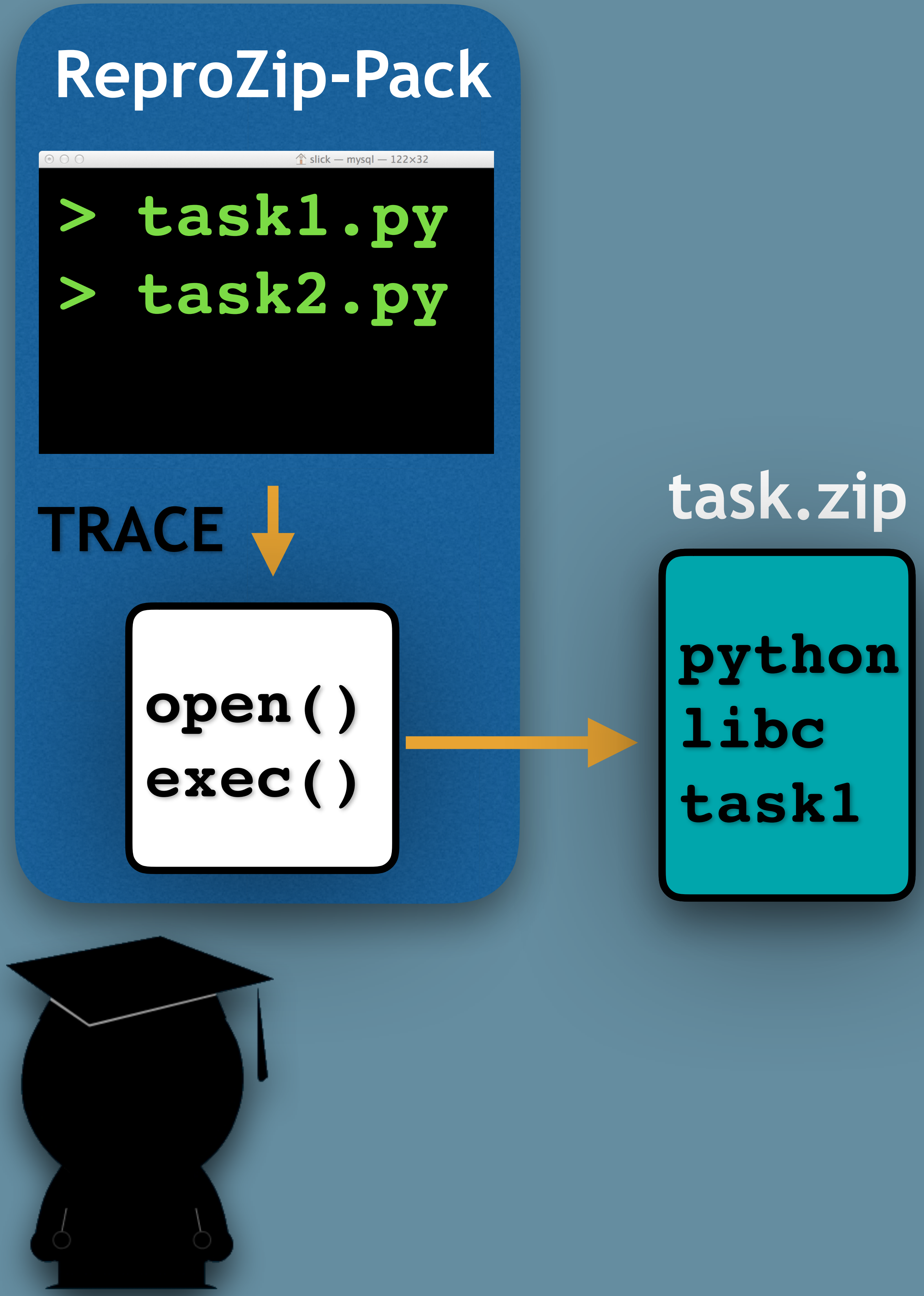
www.reprozip.org



```
> task1.py
> task2.py
```

```
open()
exec()
```

www.reprozip.org

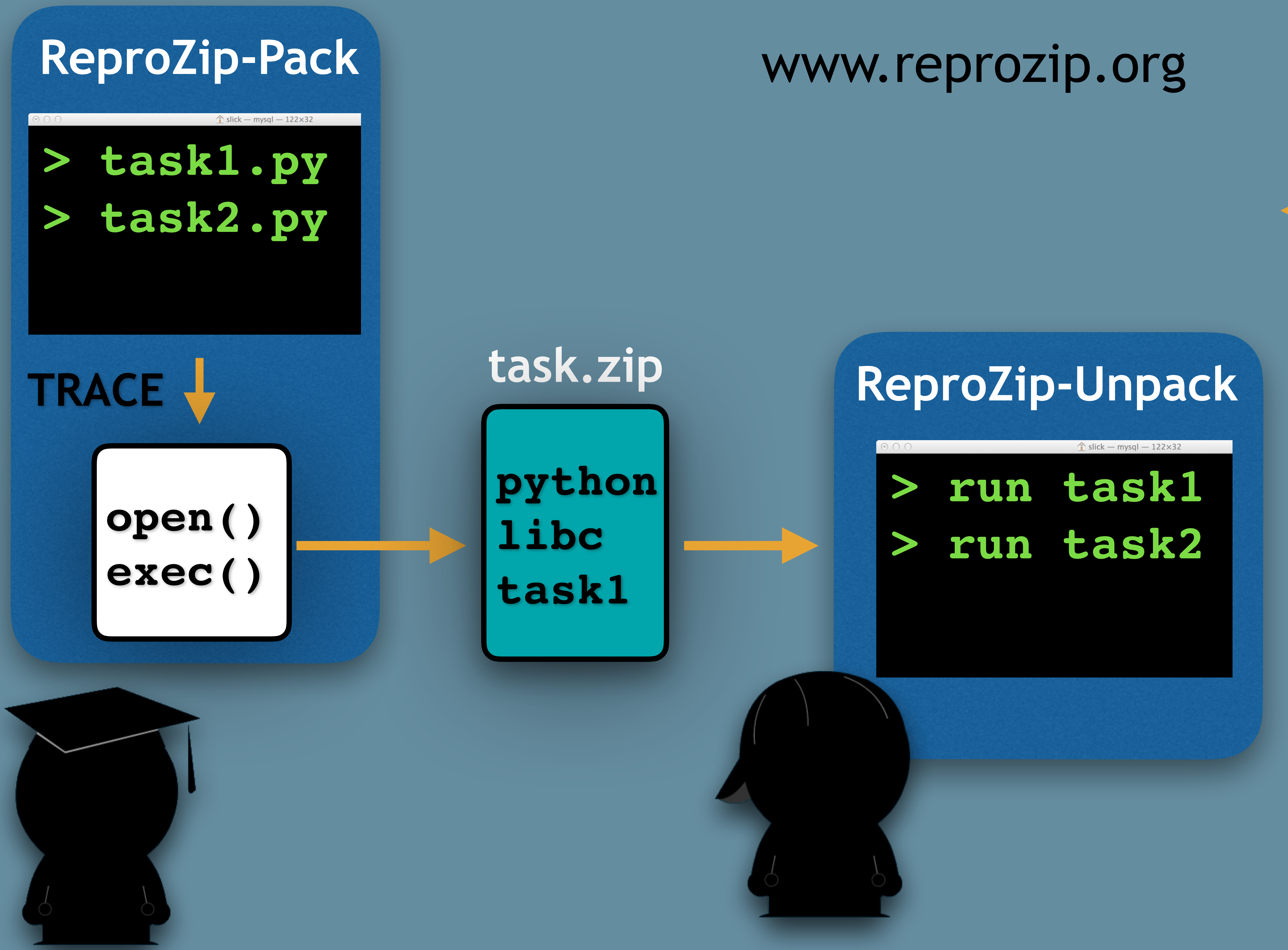


```
> task1.py
> task2.py
```

```
open()
exec()
```

```
python
libc
task1
```

www.reprozip.org



```
> task1.py
> task2.py
```

```
open()
exec()
```

```
python
libc
task1
```

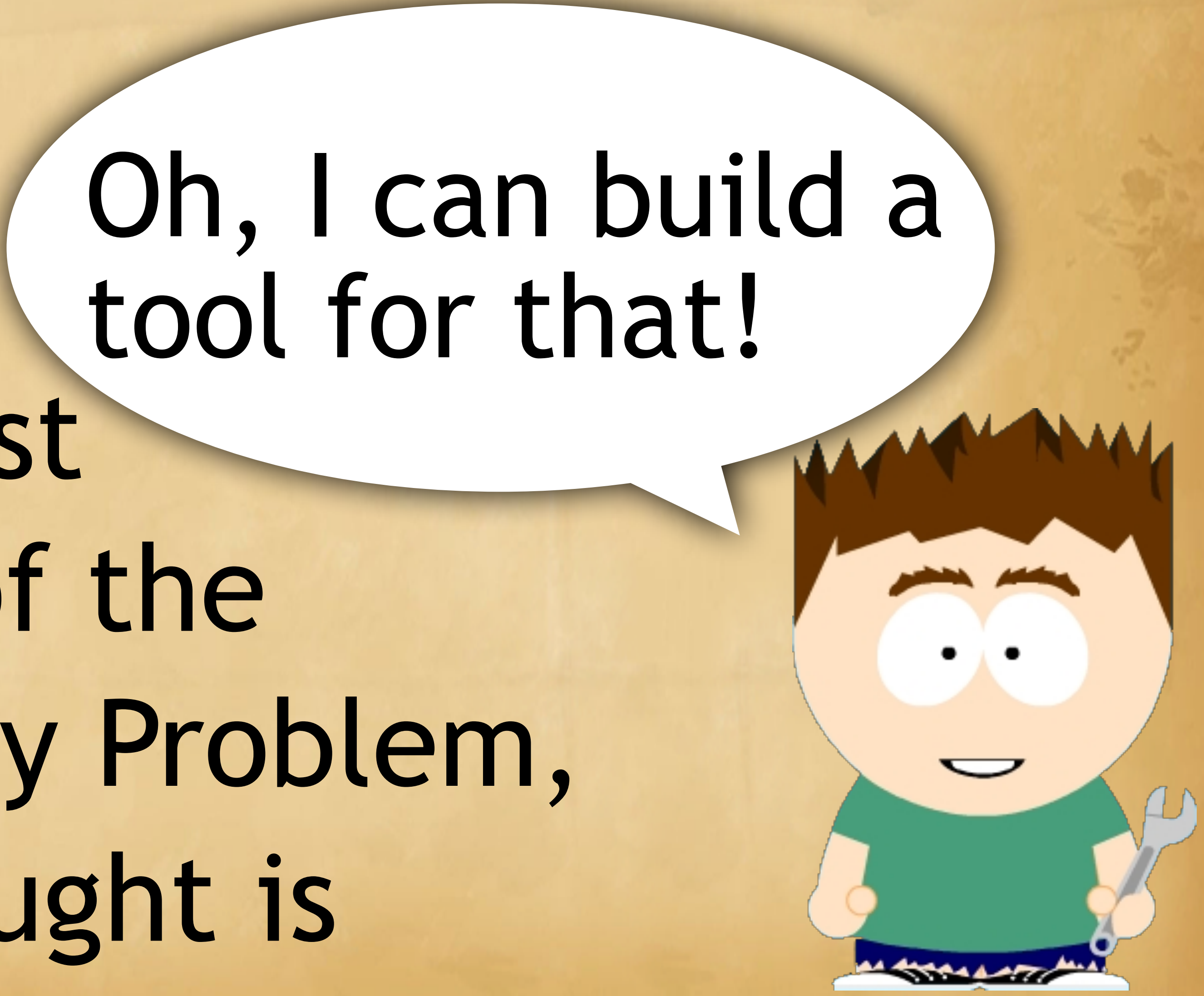
```
> run task1
> run task2
```

4th Law of Artifact Sharing

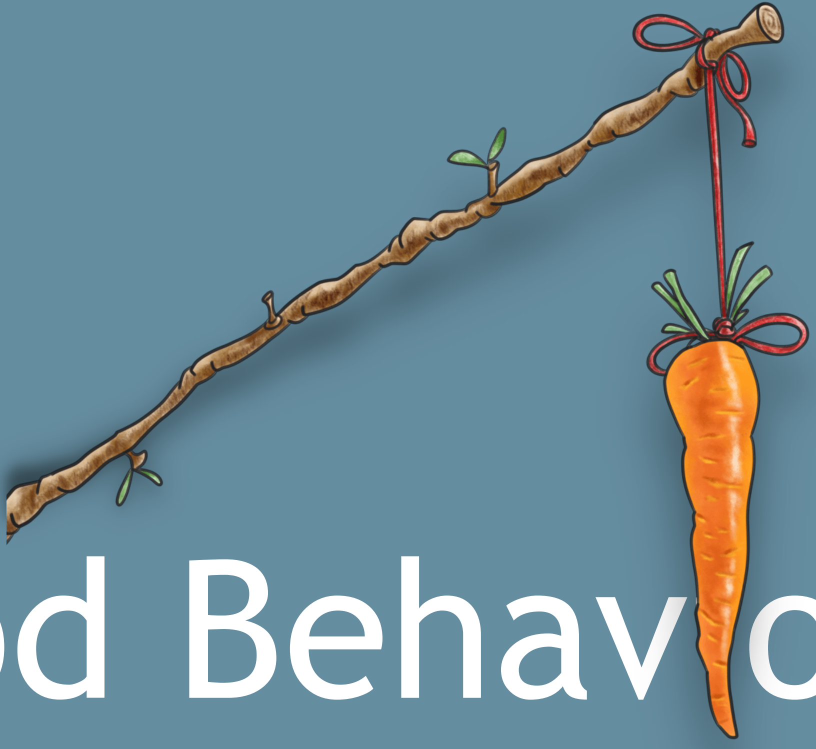
When a Computer Scientist is first made aware of the Reproducibility Problem, their first thought is

4th Law of Artifact Sharing

When a Computer Scientist is first made aware of the Reproducibility Problem, their first thought is

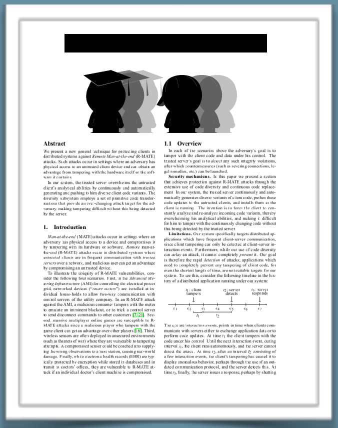


Sharing Proposal — #4 —

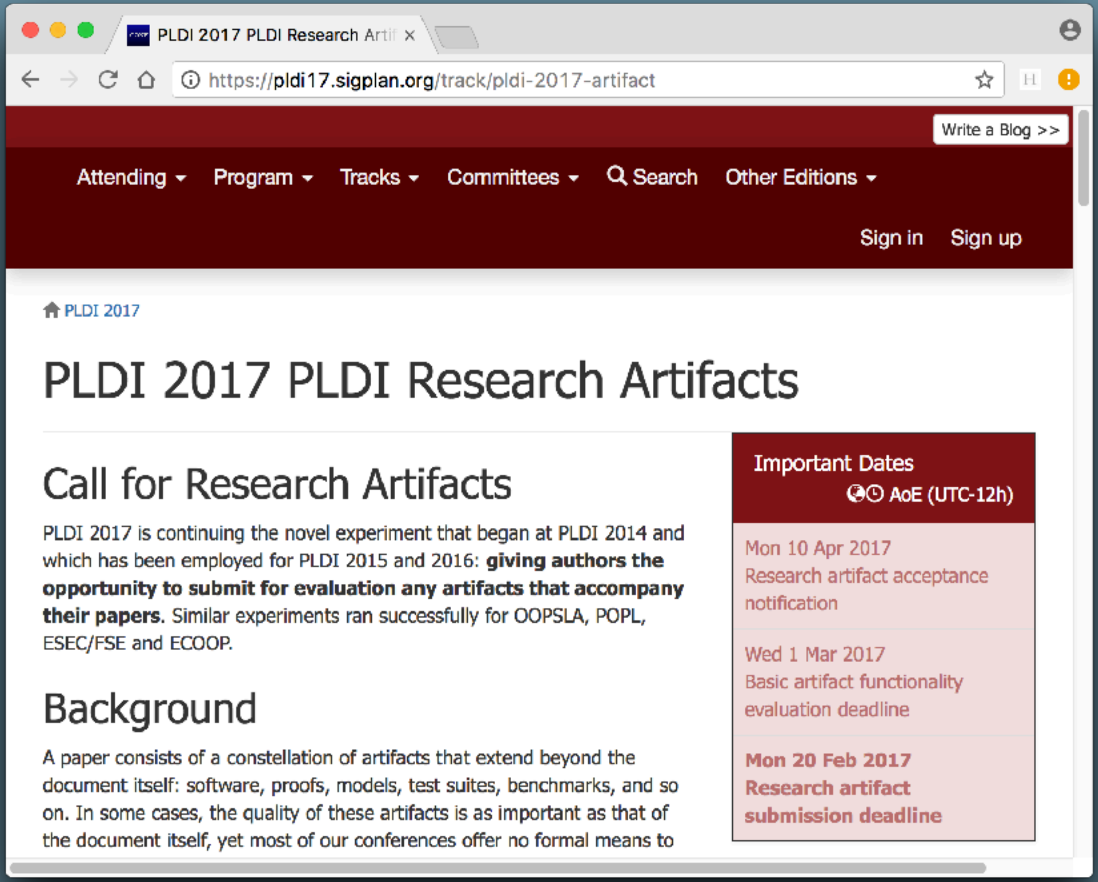


Rewarding Good Behavior

Sharing Proposal
#4 —
Rewarding Good Behavior



ARTIFACT



PLDI 2017 PLDI Research Artifacts

Call for Research Artifacts

PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: **giving authors the opportunity to submit for evaluation any artifacts that accompany their papers**. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.

Background

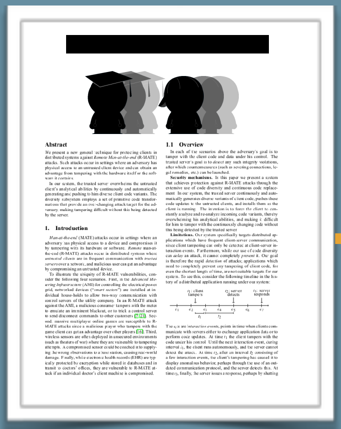
A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to

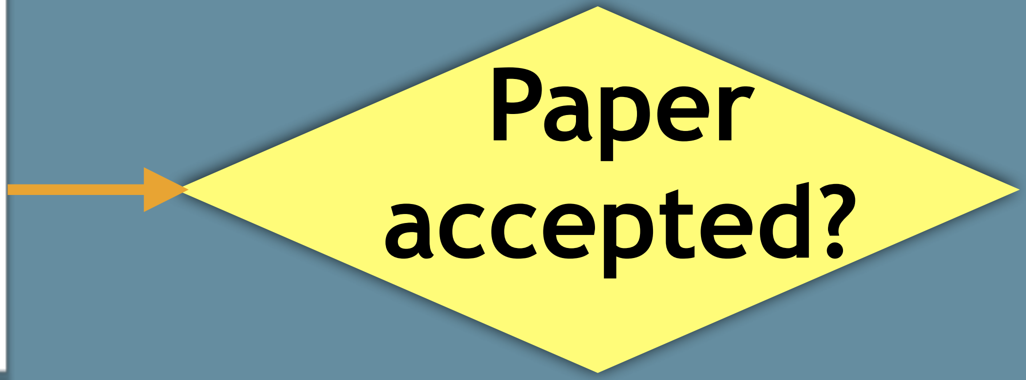
Important Dates
AoE (UTC-12h)

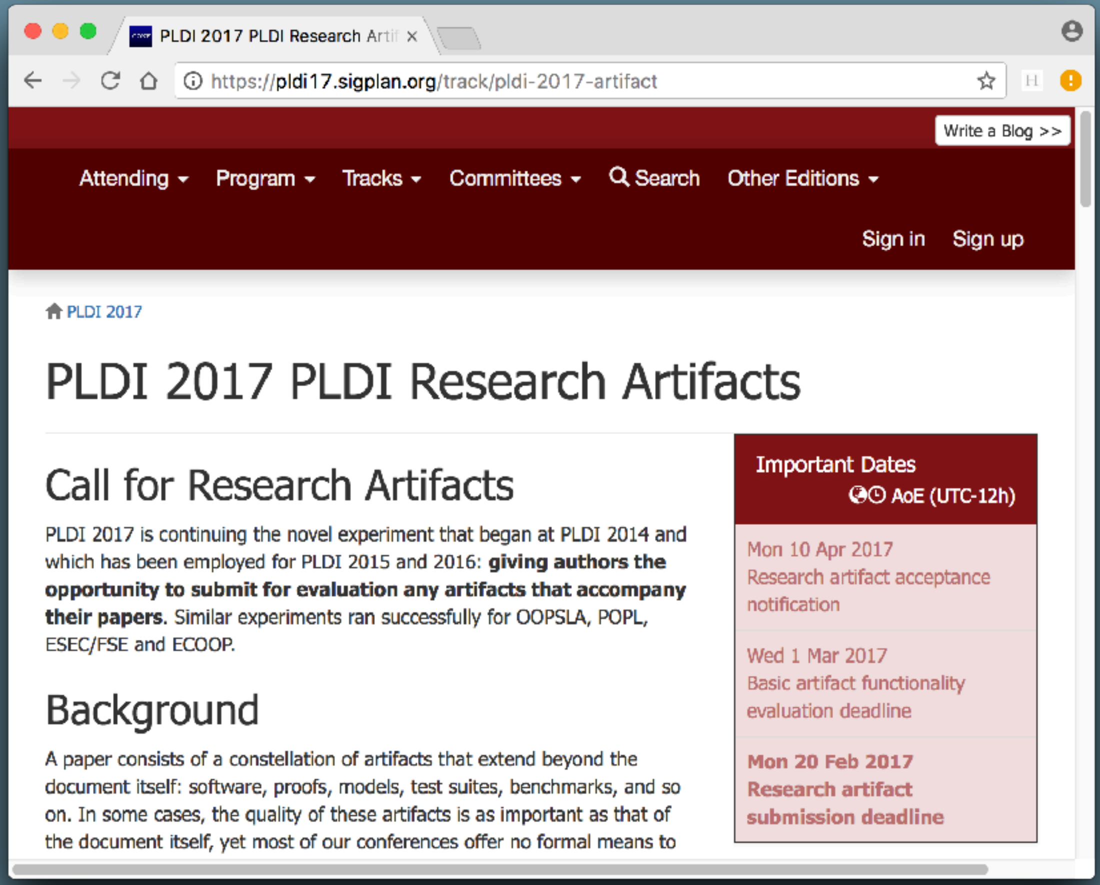
Mon 10 Apr 2017
Research artifact acceptance notification

Wed 1 Mar 2017
Basic artifact functionality evaluation deadline

Mon 20 Feb 2017
Research artifact submission deadline







PLDI 2017 PLDI Research Artifacts

Call for Research Artifacts

PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: **giving authors the opportunity to submit for evaluation any artifacts that accompany their papers**. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.

Background

A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to

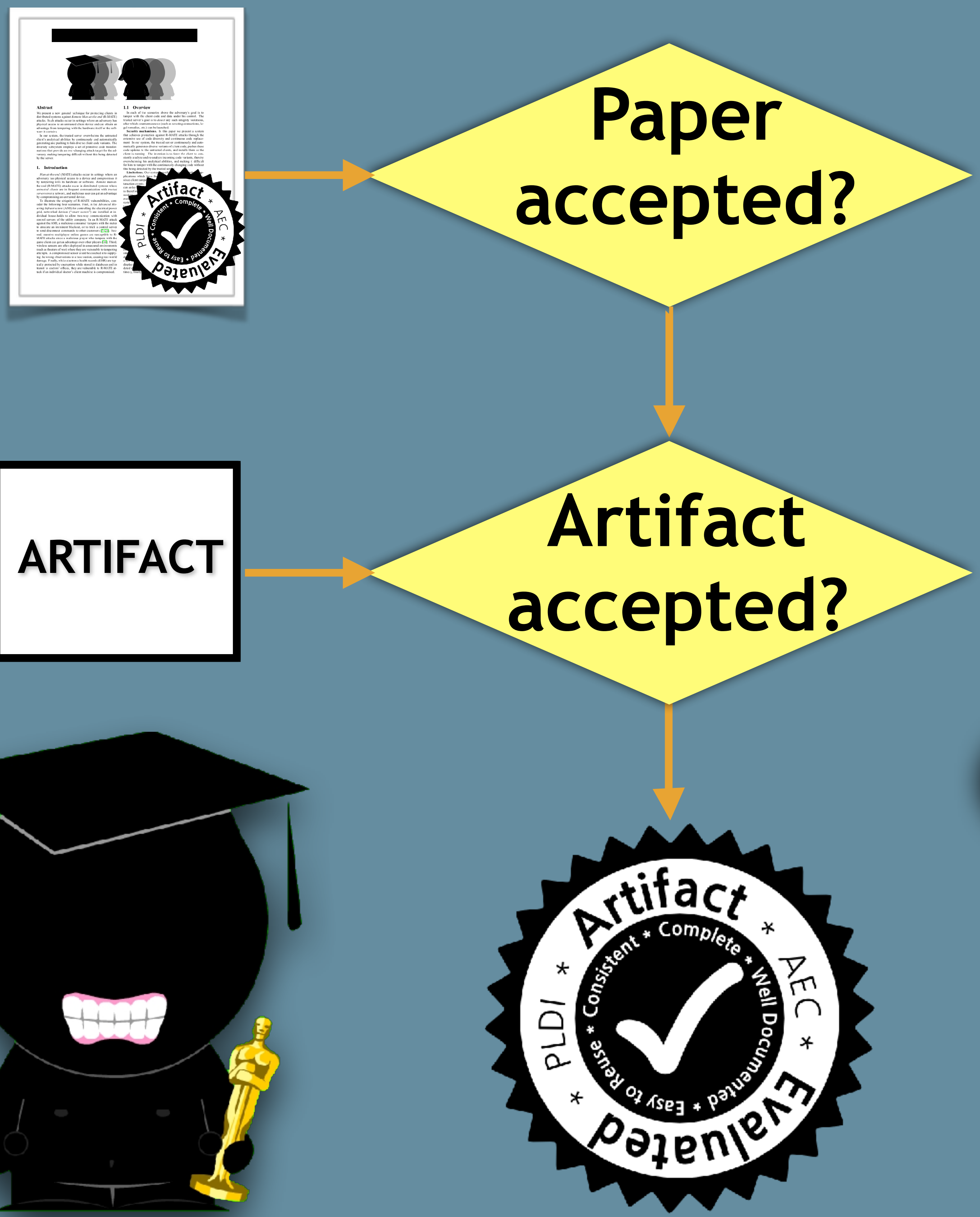
Important Dates
AoE (UTC-12h)

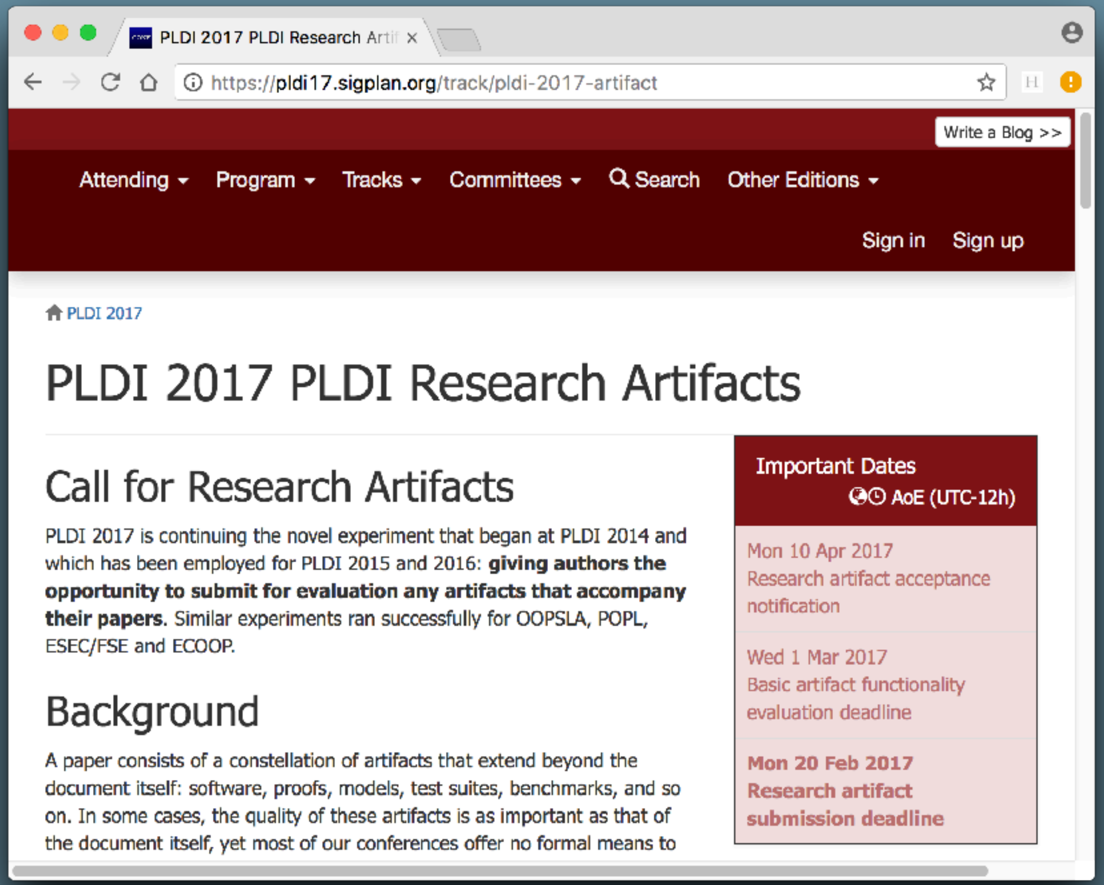
Mon 10 Apr 2017
Research artifact acceptance notification

Wed 1 Mar 2017
Basic artifact functionality evaluation deadline

Mon 20 Feb 2017
Research artifact submission deadline

ARTIFACT




PLDI 2017 PLDI Research Arti

https://pldi17.sigplan.org/track/pldi-2017-artifact

Write a Blog >>

Attending Program Tracks Committees Search Other Editions

Sign in Sign up

PLDI 2017

PLDI 2017 PLDI Research Artifacts

Call for Research Artifacts

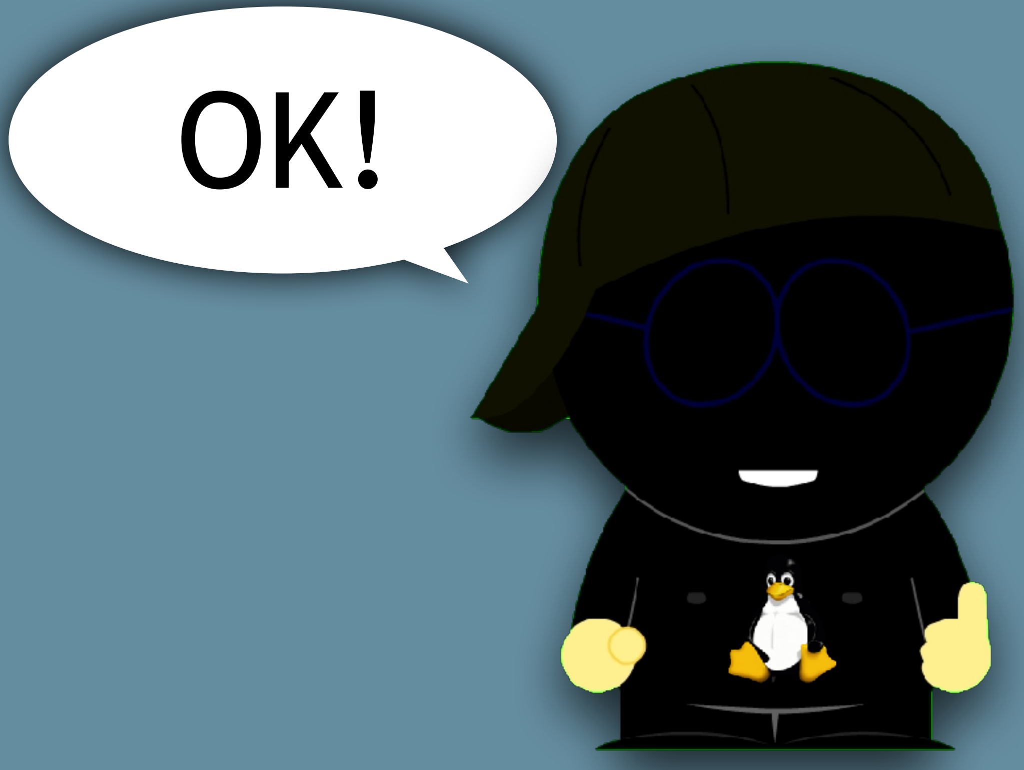
PLDI 2017 is continuing the novel experiment that began at PLDI 2014 and which has been employed for PLDI 2015 and 2016: **giving authors the opportunity to submit for evaluation any artifacts that accompany their papers**. Similar experiments ran successfully for OOPSLA, POPL, ESEC/FSE and ECOOP.

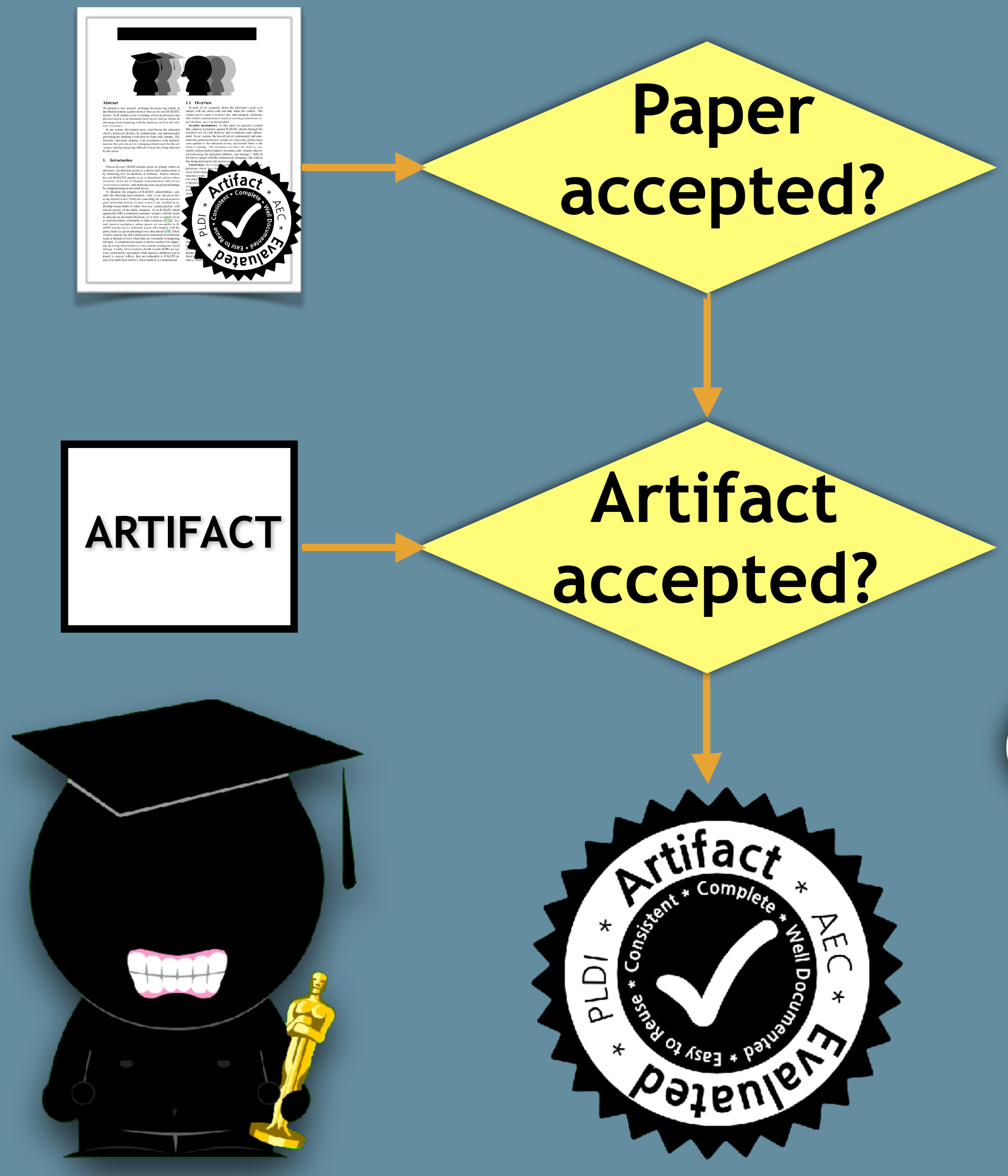
Background

A paper consists of a constellation of artifacts that extend beyond the document itself: software, proofs, models, test suites, benchmarks, and so on. In some cases, the quality of these artifacts is as important as that of the document itself, yet most of our conferences offer no formal means to

Important Dates
AoE (UTC-12h)

Mon 10 Apr 2017	Research artifact acceptance notification
Wed 1 Mar 2017	Basic artifact functionality evaluation deadline
Mon 20 Feb 2017	**Research artifact submission deadline**

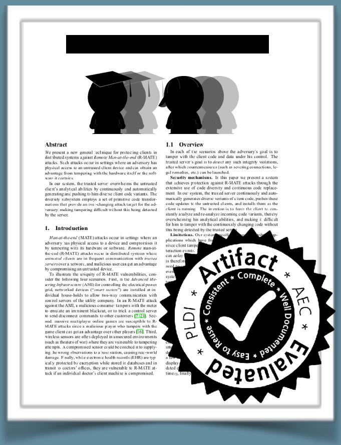




- Voluntary
- Does not affect accept/reject
- No expectation of sharing

Artifact
Consistent • Complete • Well Documented • Easy to Reuse
Evaluated
AEC
PLDI

ARTIFACT



ARTIFACT

Repeatability

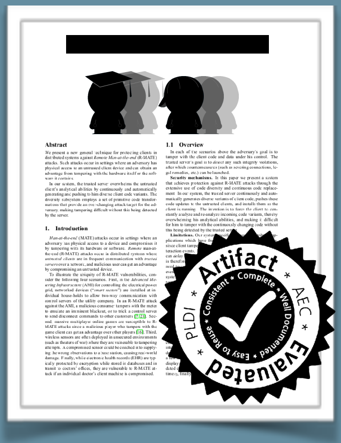
ARTIFACT

Repeatability

Results Replicated acm

Reproducibility

Results Reproduced acm



ARTIFACT

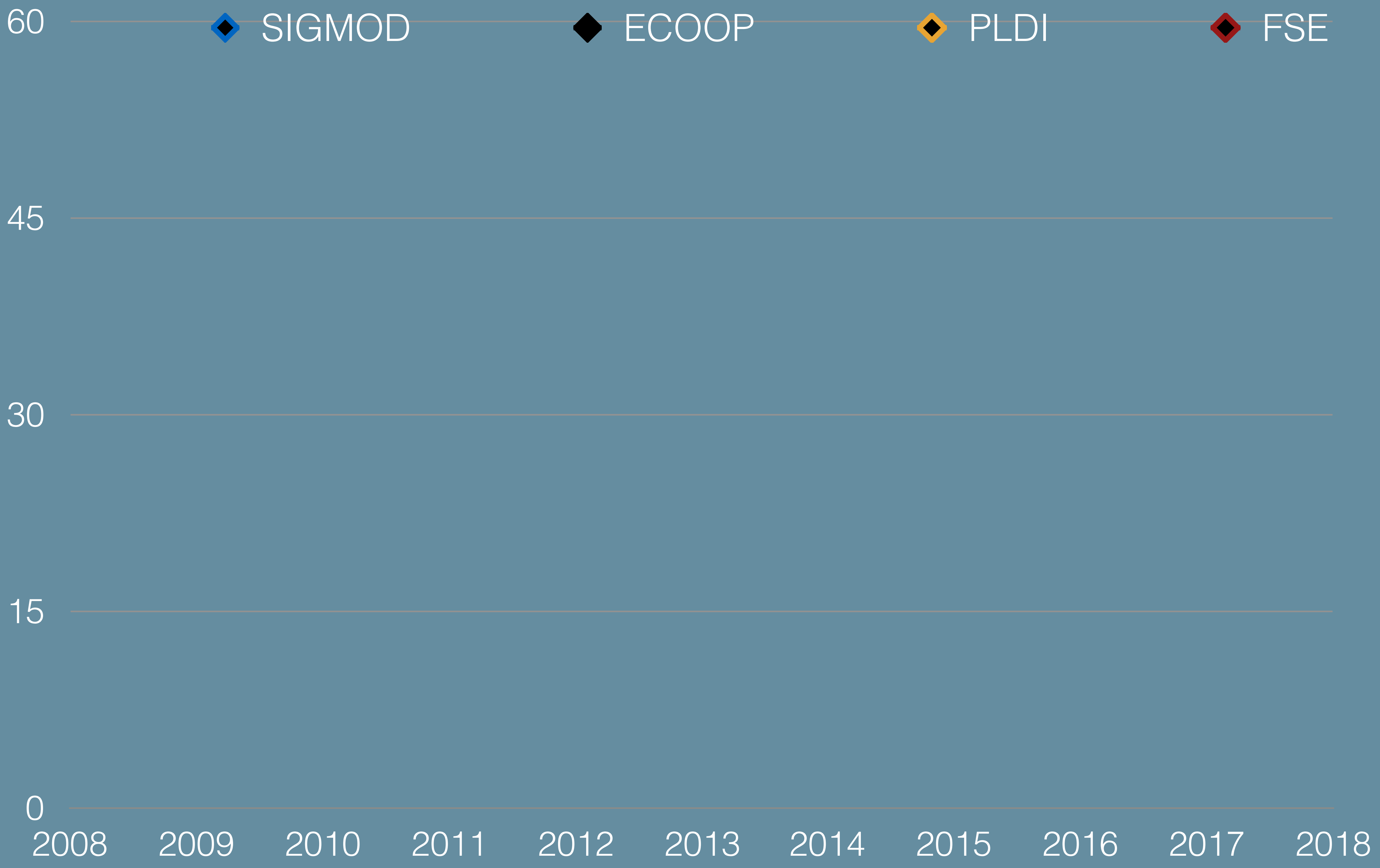
Repeatability

Reproducibility

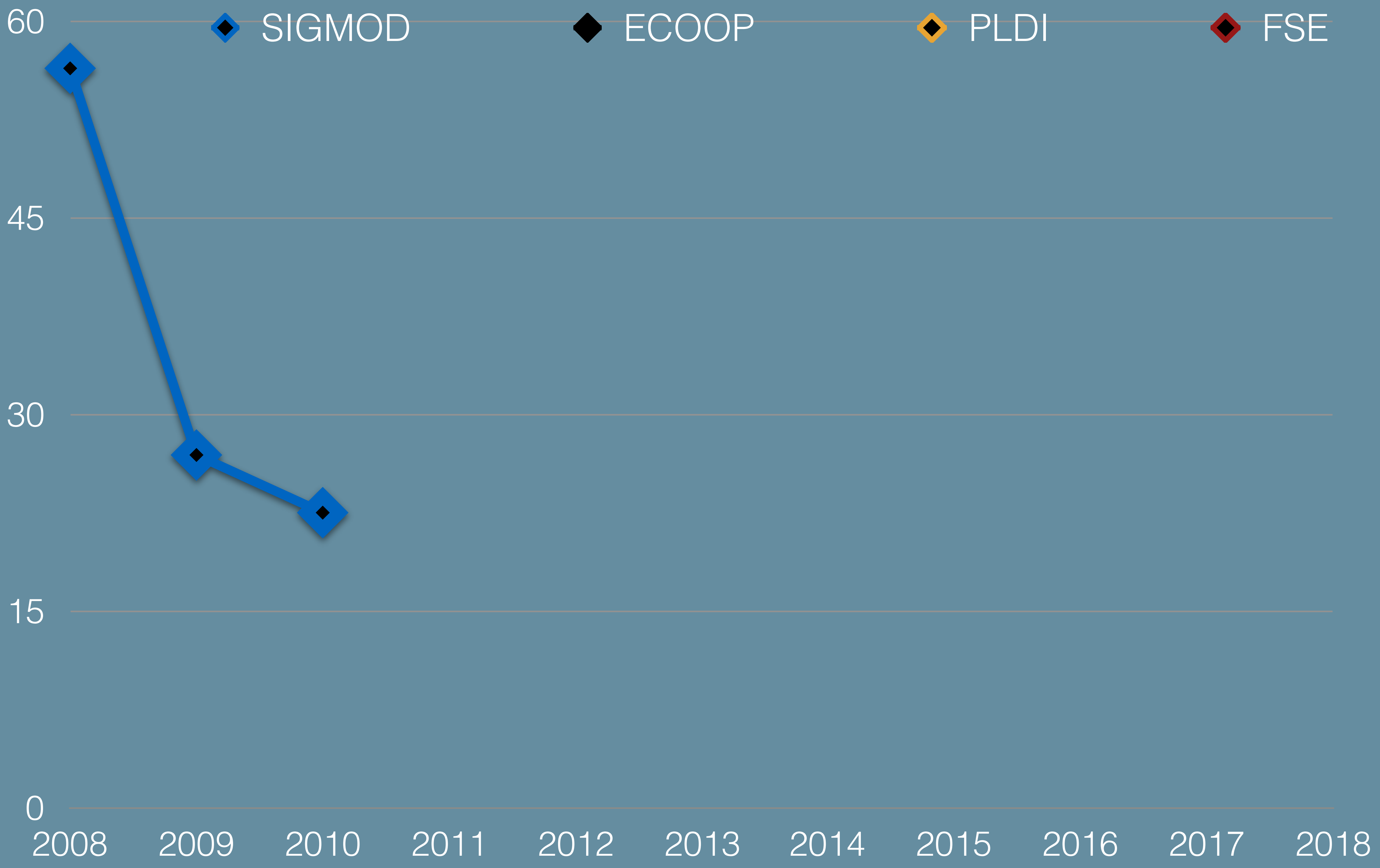
Benefaction

Accepted Artifacts / Accepted Papers (%)

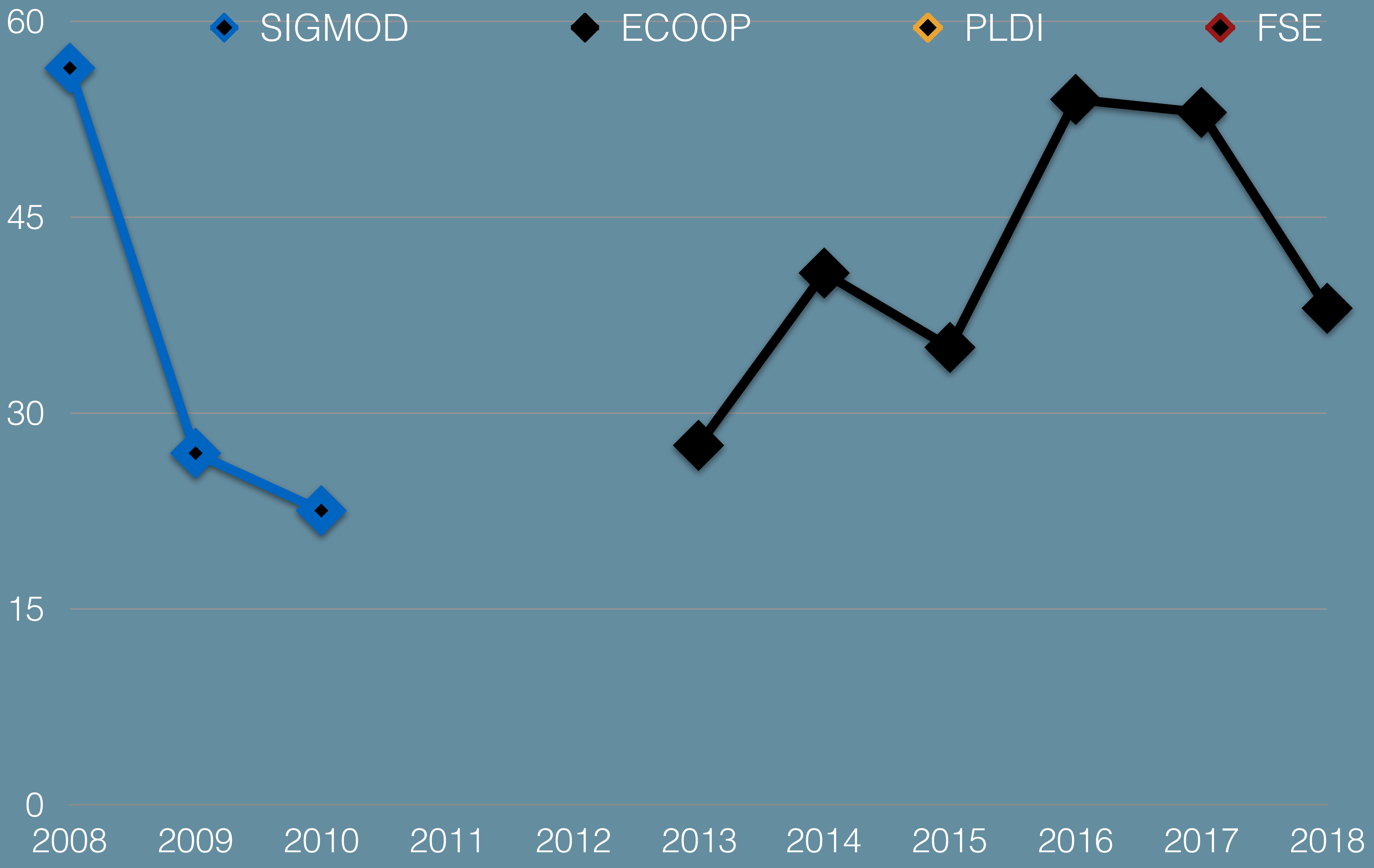
Accepted Artifacts / Accepted Papers (%)



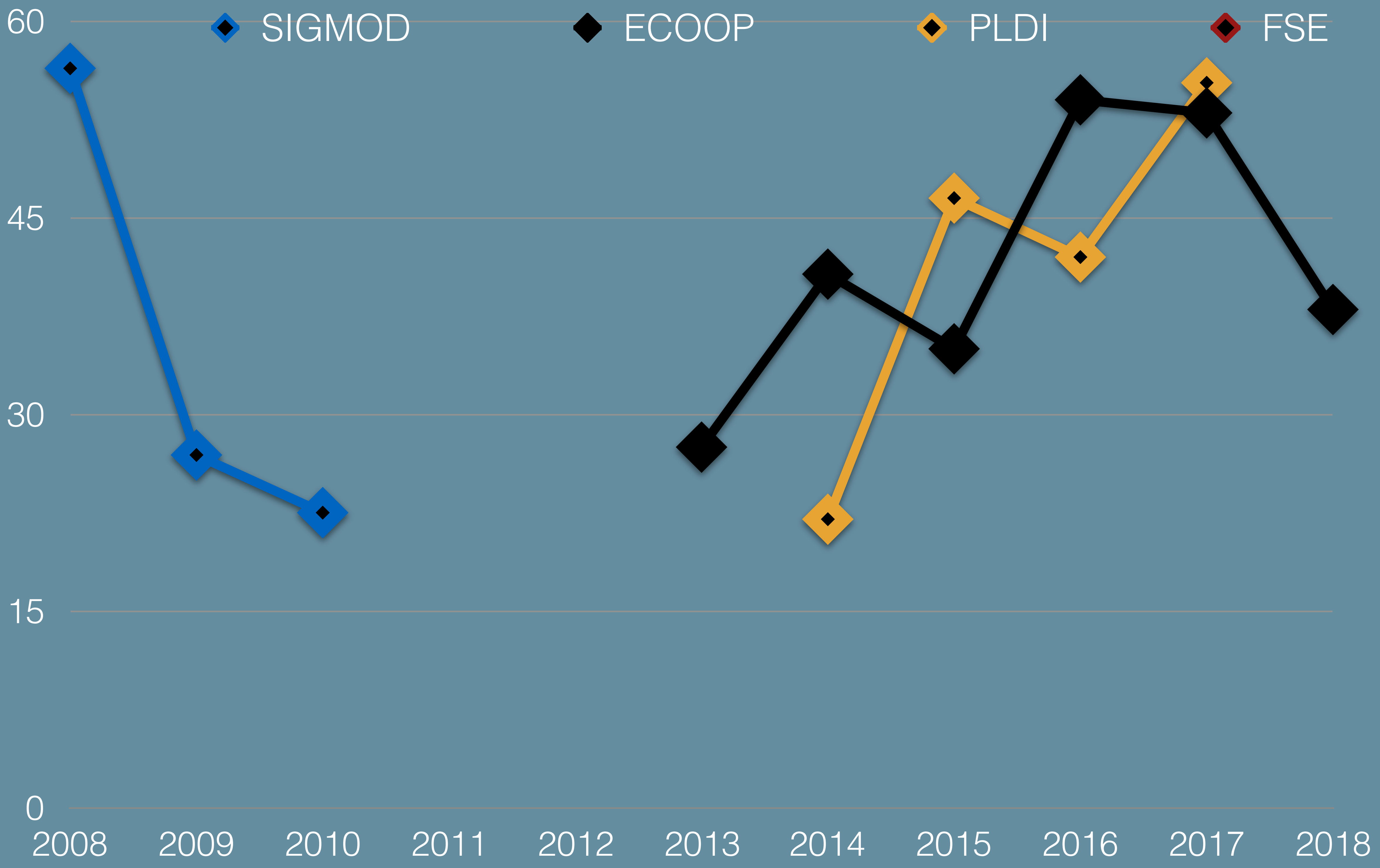
Accepted Artifacts / Accepted Papers (%)



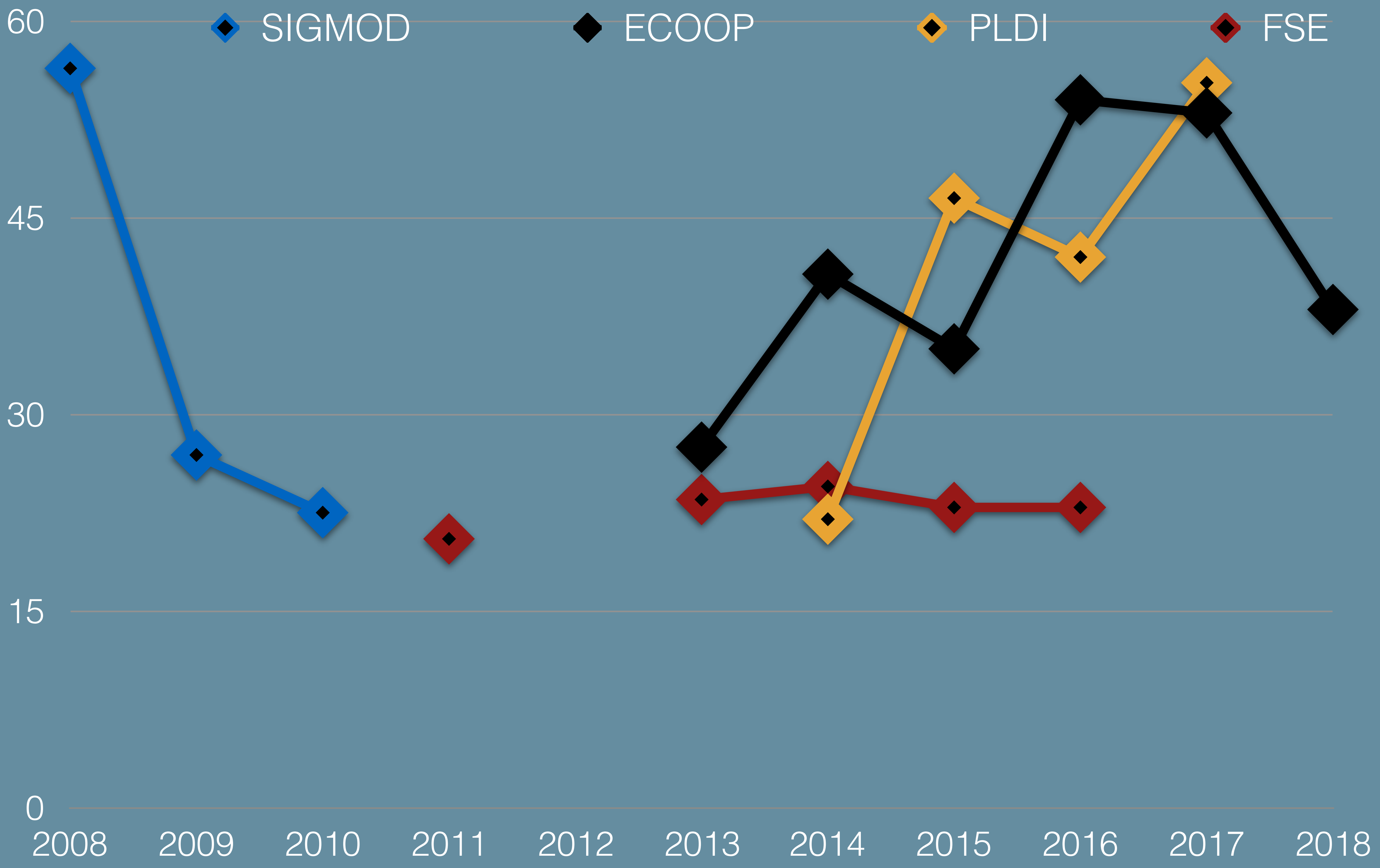
Accepted Artifacts / Accepted Papers (%)



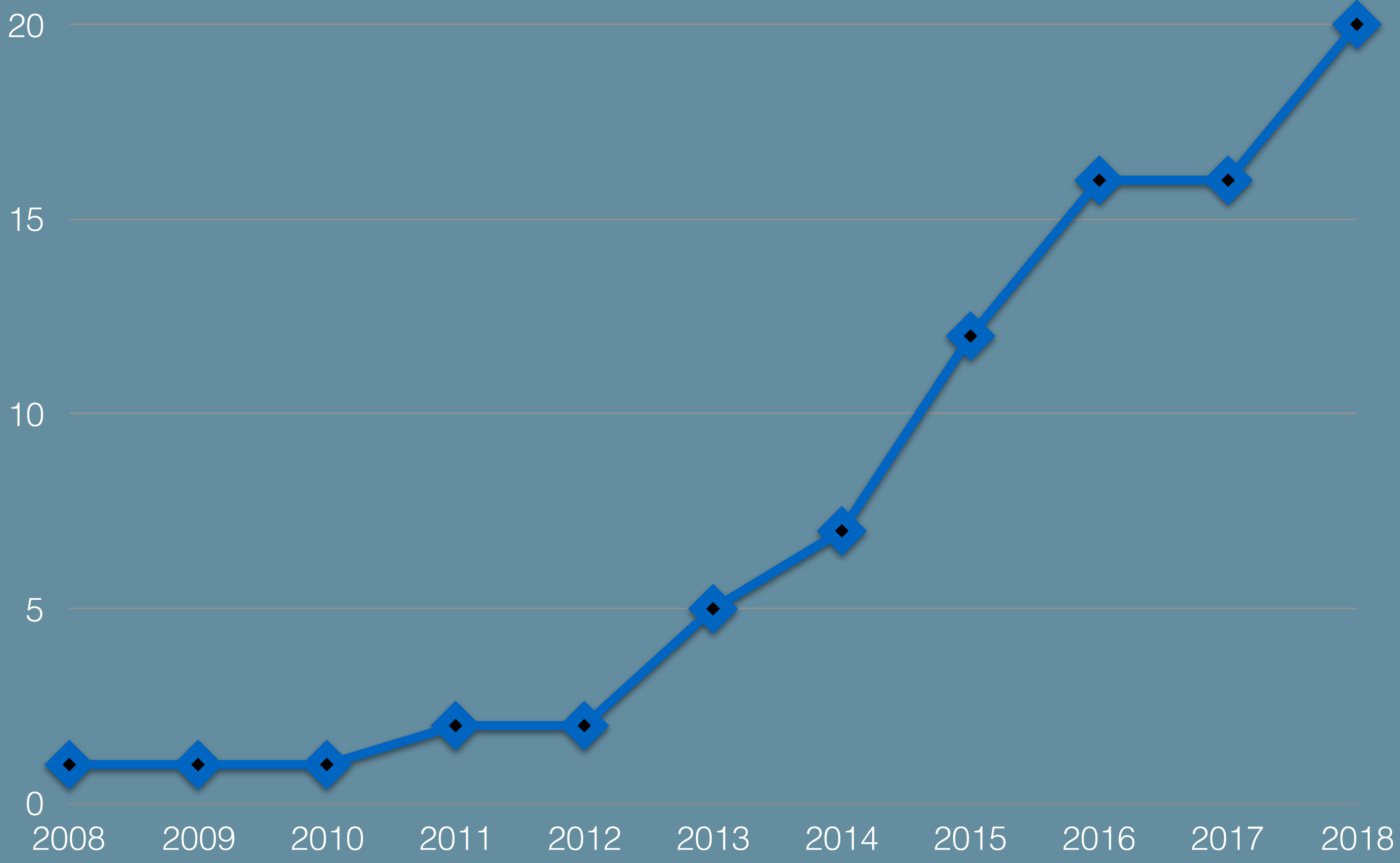
Accepted Artifacts / Accepted Papers (%)



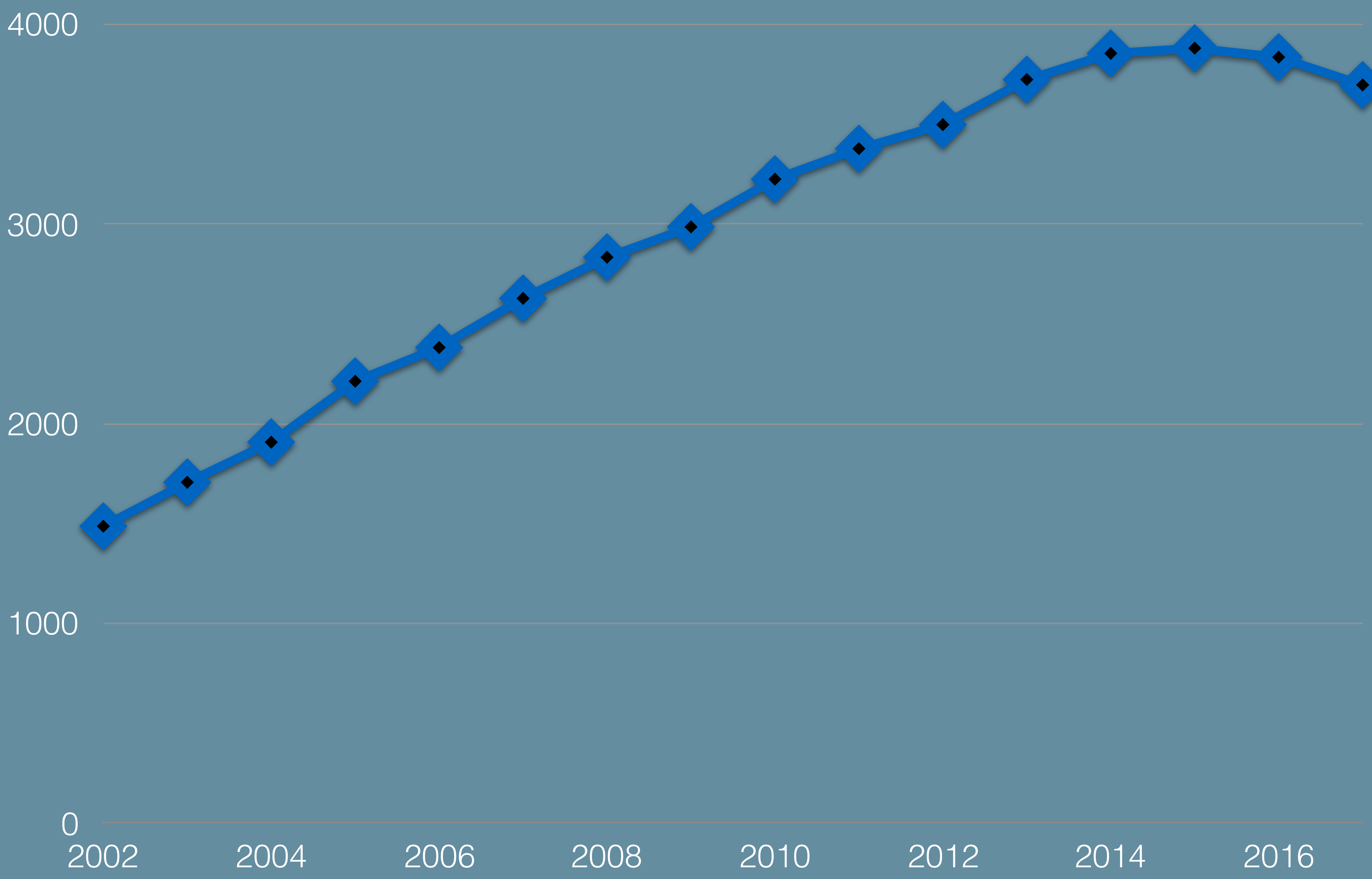
Accepted Artifacts / Accepted Papers (%)



# Conferences with AE



Publication Venues (Dblp)

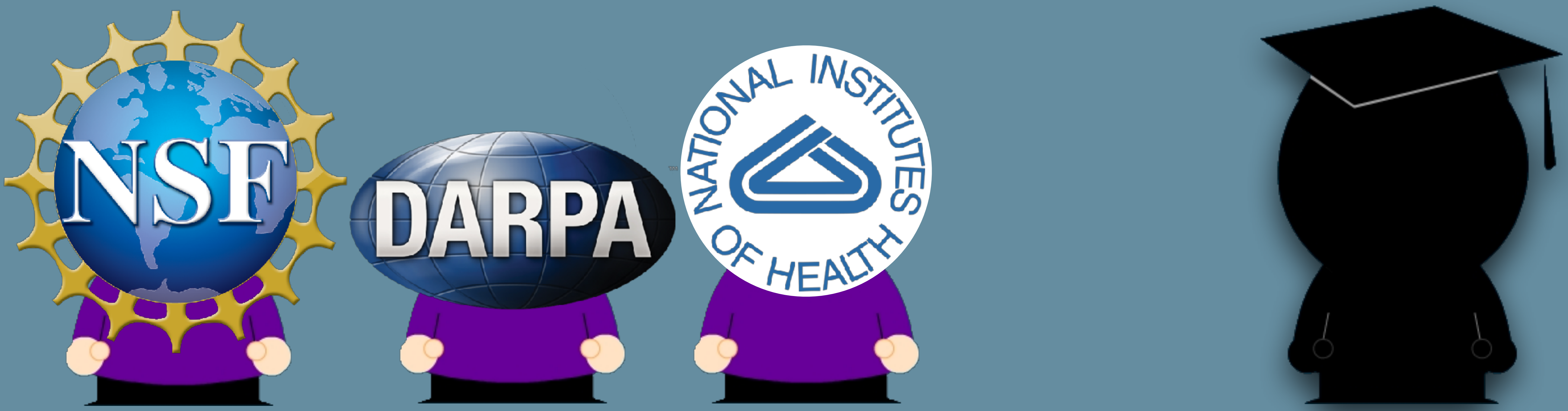


Sharing Proposal — #5 —

Punishing Bad Behavior

Grant application #: [redacted]

We will make our data and software available.

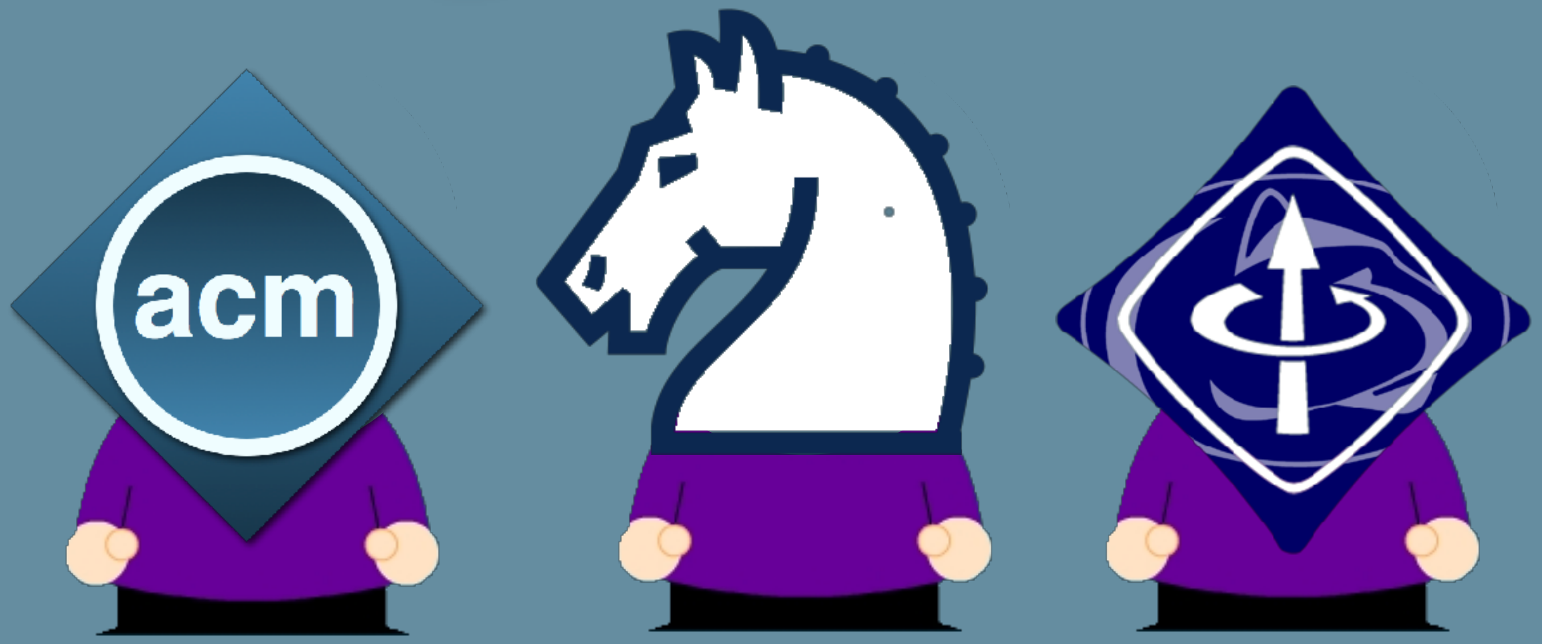


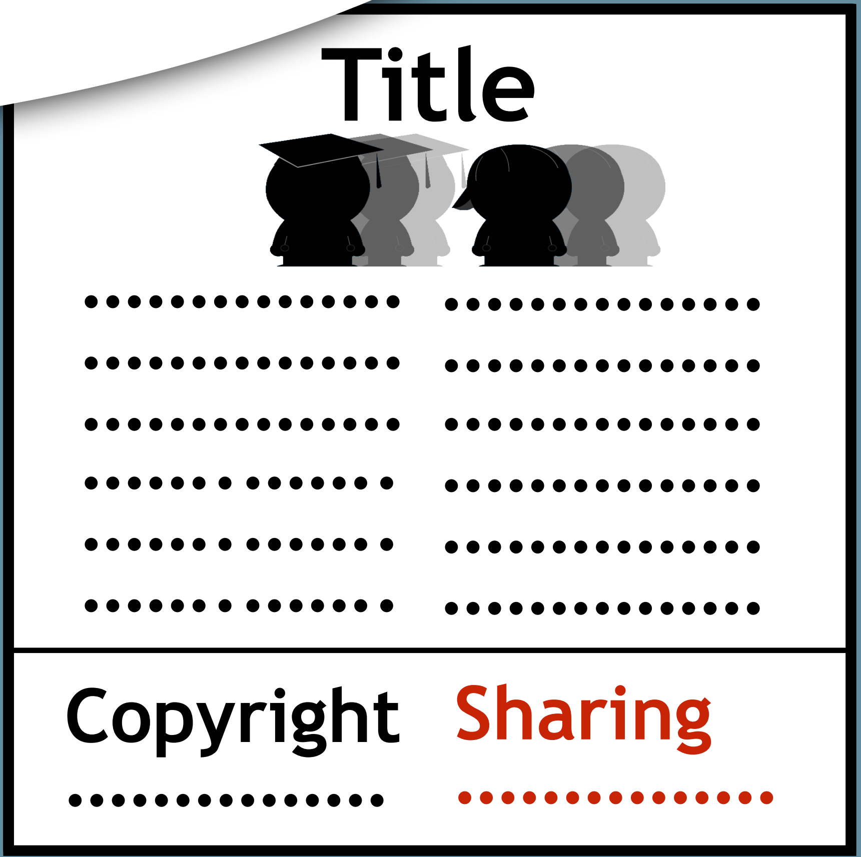
Random Audit!

Are you sharing like you promised in the grant application?

Sharing Contract!

What level of sharing are you committing to?

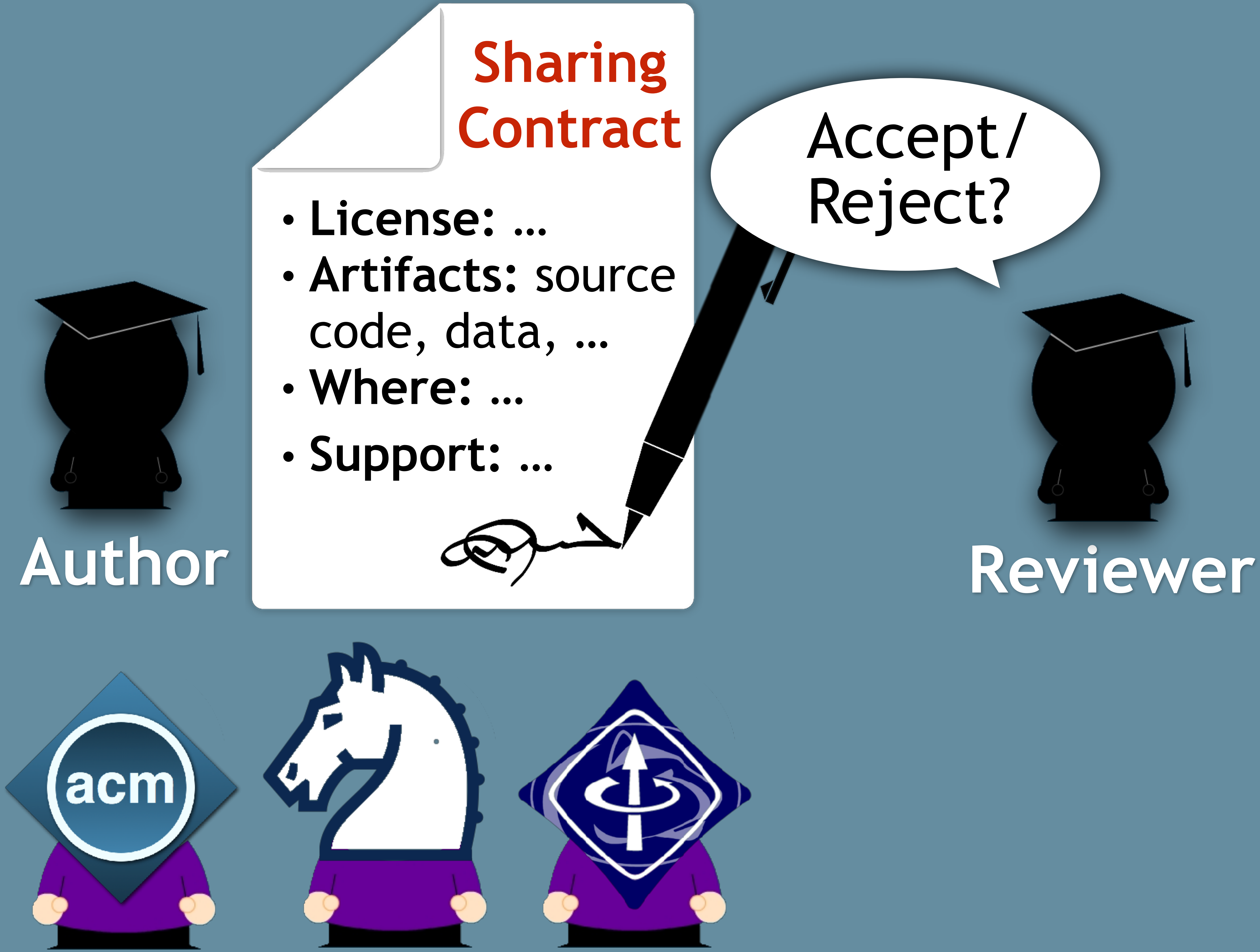




Sharing Contract

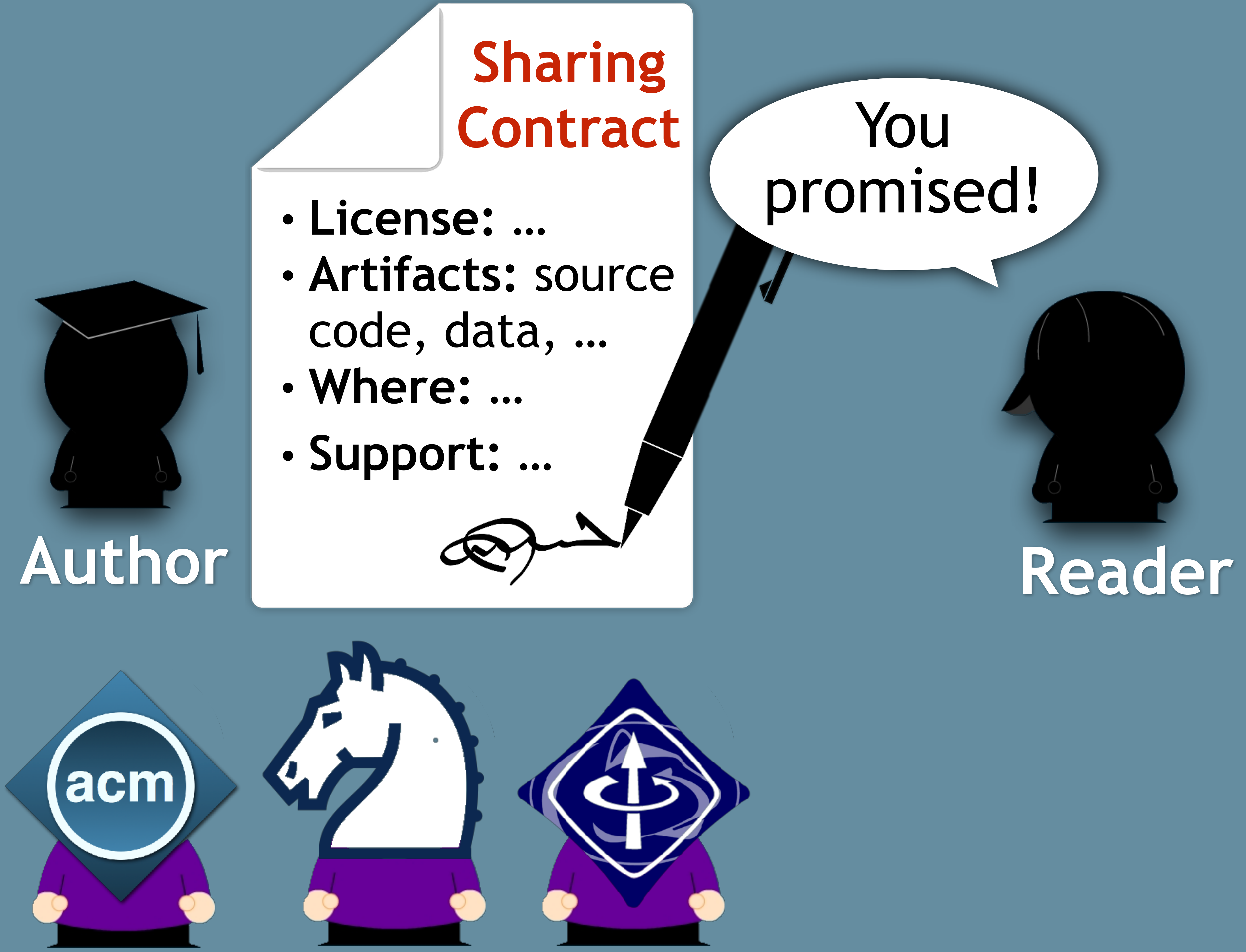
- **License:** …
- **Artifacts:** source code, data, …
- **Where:** …
- **Support:** …

[illegible]



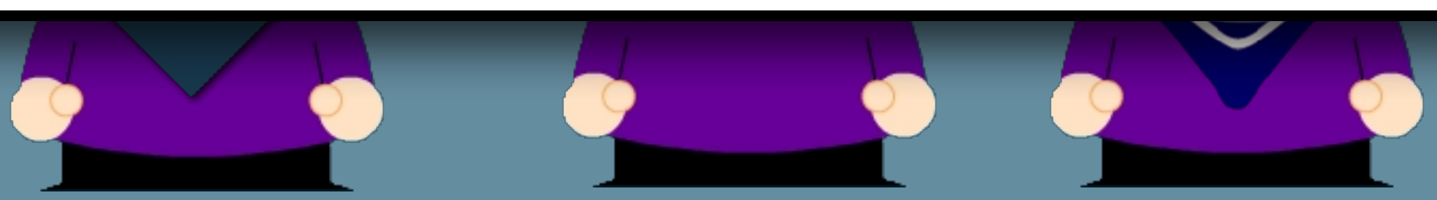
Sharing Contract

- License: …
- Artifacts: source code, data, …
- Where: …
- Support: …


Sharing Contract
• License: …
• Artifacts: source code, data, …
• Where: …
• Support: …
You promised!
Author
Reader
acm

nature

a statement must be included … indicating … how the code … can be accessed, including any restrictions …

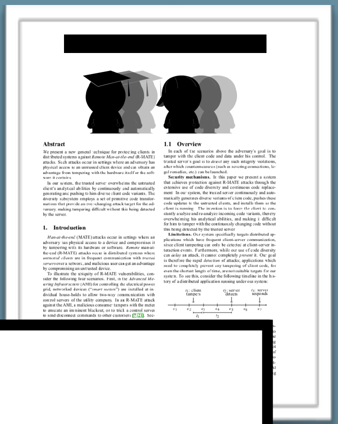


Sharing Proposal — #6 —

Optional Sharing

Sharing Proposal — #6 —

Mandated Sharing



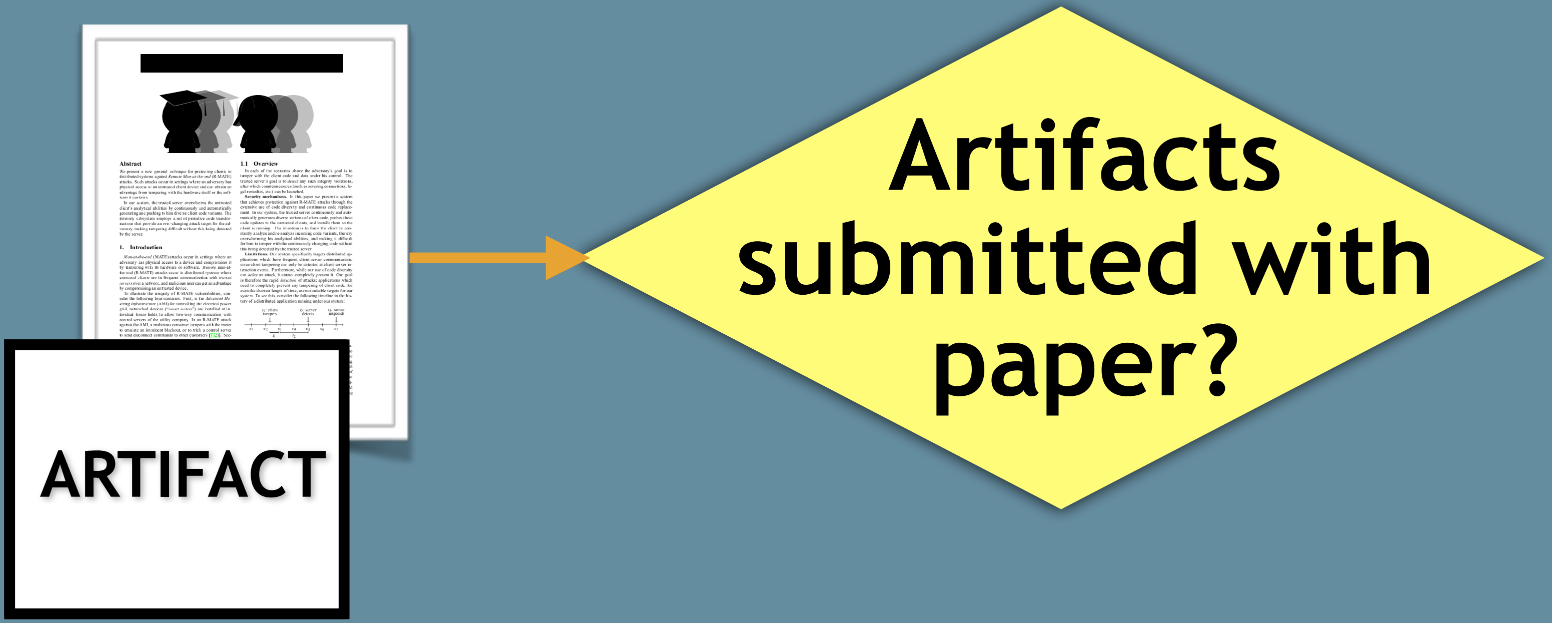
ARTIFACT

Volume 2, No. 1

March 1975

JOURNAL OF Development ECONOMICS

CONTENTS



Volume 2, No. 1
March 1975

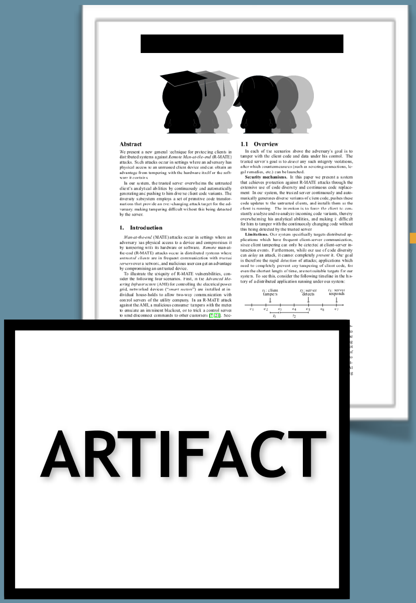
JOURNAL OF Development ECONOMICS

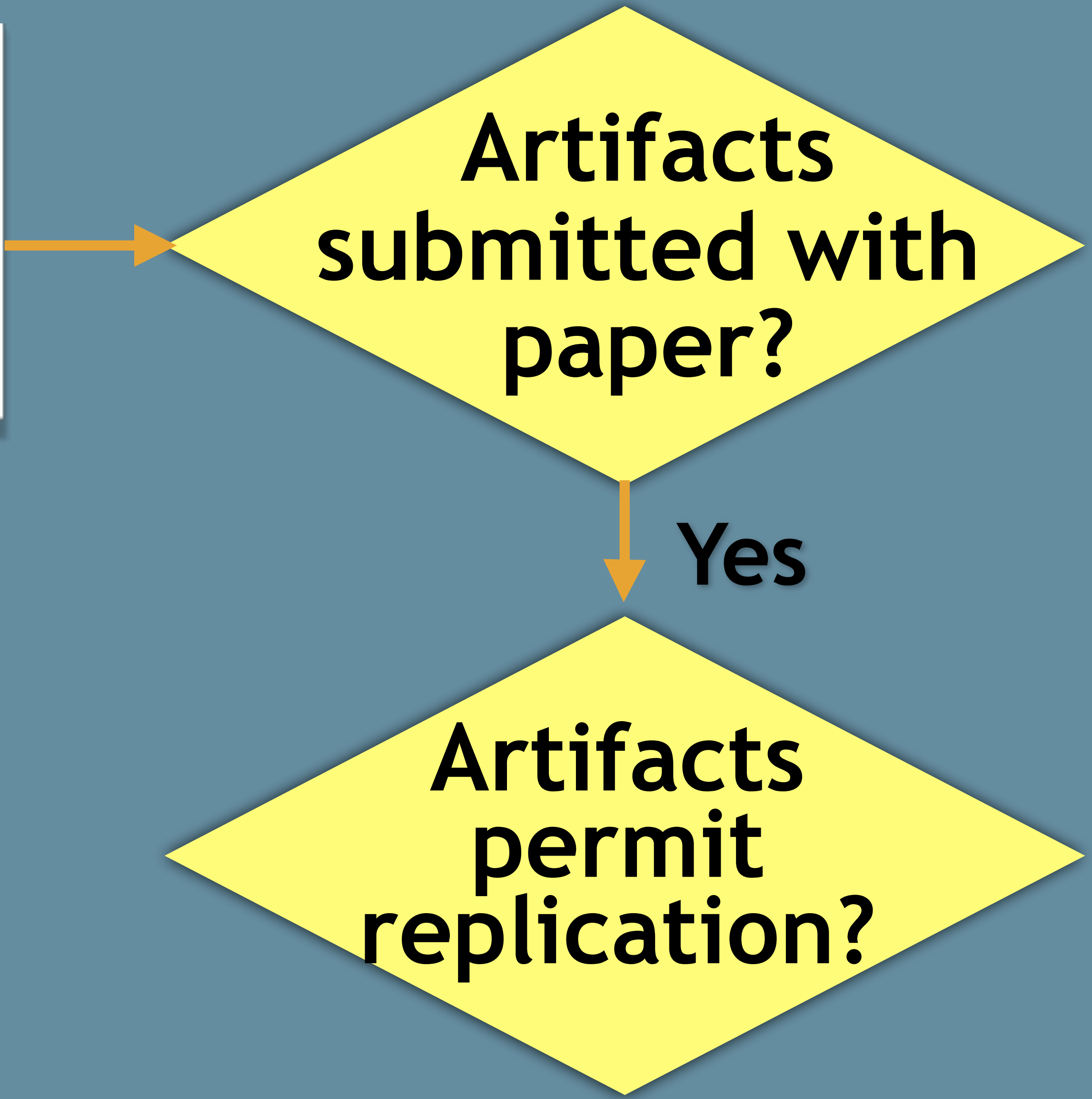
Editors:
ALAN S. MANNE
T.N. SRINIVASAN

Review Editor:
PADMA DESAI

Associate Editors:
I. ADELMAN
E.L. BACHA
J.N. BHAGWATI
M. BRUNO
H.J. BRUTON
M. CELASUN
P. DASGUPTA
DIAZ ALEJANDRO
J. ENCARNACION
E. GARCIA
M. GARGOURI
D. GHAI
R. JOLLY
J.M. KATZ
L.E.M. MENNES
Y. MURAKAMI
H.M.A. ONITIRI
G. PAPANDREOU
J. SANDEE
L. SOLIS
L.G. STOLERU
L.J. TAYLOR
T.E. WEISSKOPF

CONTENTS




Artifacts submitted with paper?
Yes
Artifacts permit replication?

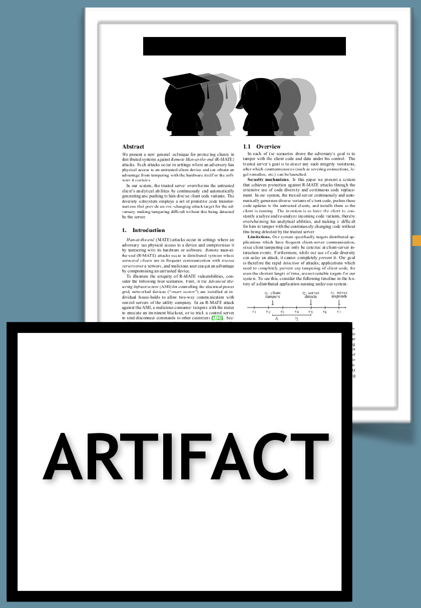
Volume 2, No. 1
March 1975

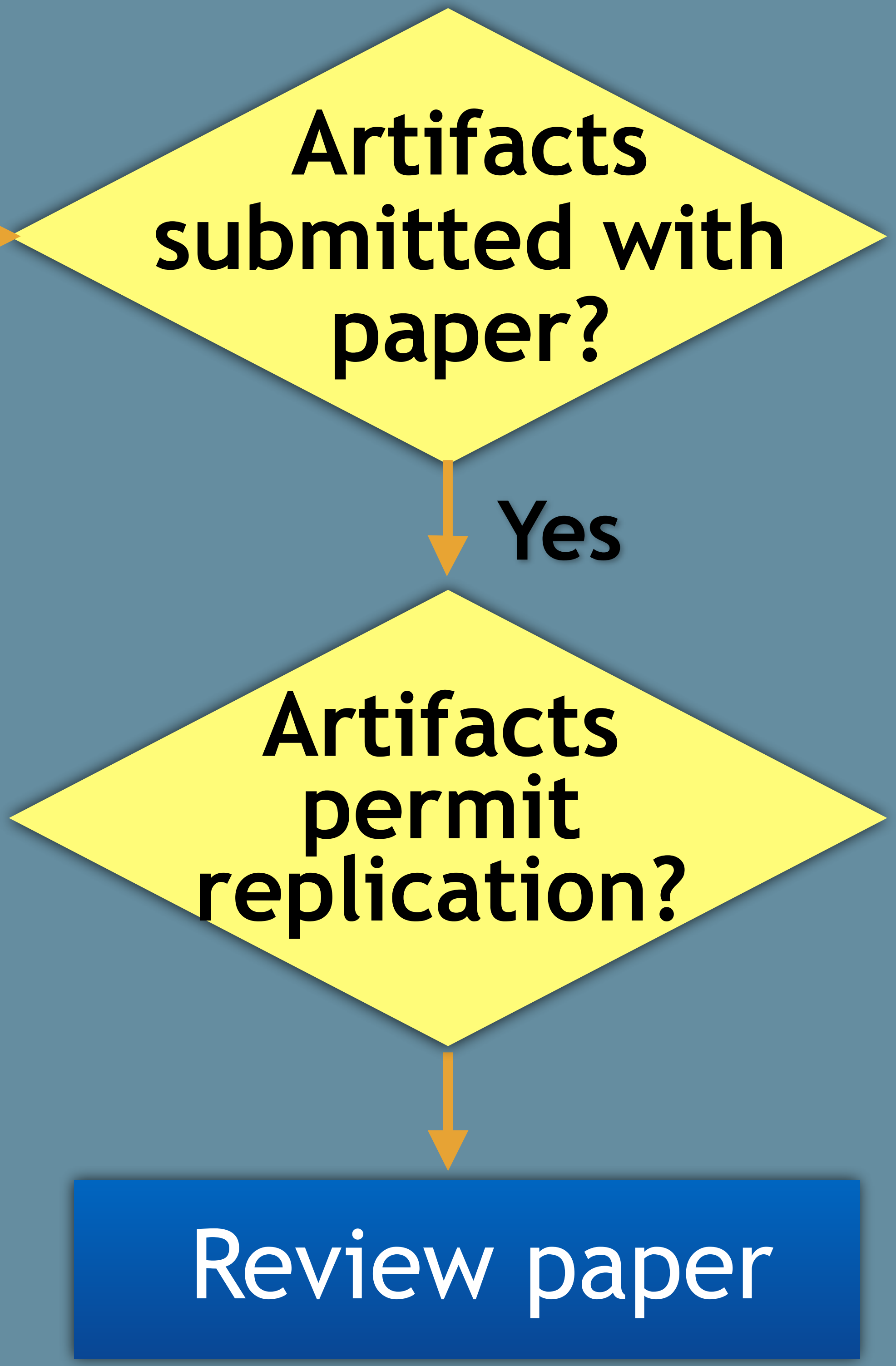
JOURNAL OF Development ECONOMICS

Editors:
ALAN S. MANNE
T.N. SRINIVASAN

Review Editor:
PADMA DESAI

sociate Editors:
I. ADELMAN
E.L. BACHA
J.N. BHAGWATI
M. BRUNO
H.J. BRUTON
M. CELASUN
P. DASGUPTA
IAZ ALEJANDRO
. ENCARNACION
E. GARCIA
M. GARGOURI
D. GHAI
R. JOLLY
J.M. KATZ
L.B.M. MENNES
Y. MURAKAMI
H.M.A. ONITIRI
G. PAPANDREOU
J. SANDEE
L. SOLIS
L.G. STOLERU
L.J. TAYLOR
T.E WEISSKOPF




Artifacts submitted with paper?
Yes
Artifacts permit replication?
Review paper

Volume 2, No. 1
March 1975

JOURNAL OF Development ECONOMICS

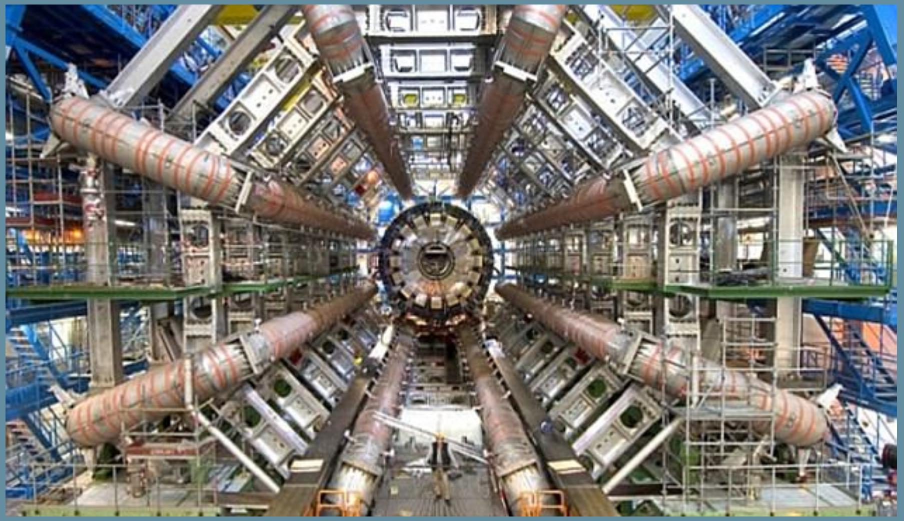
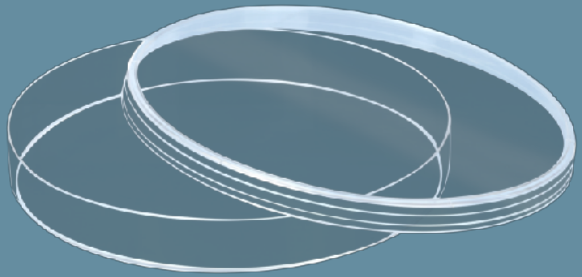
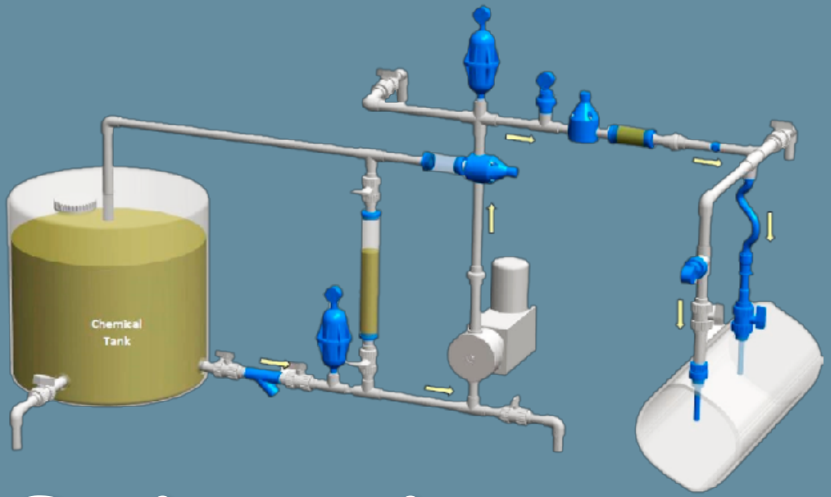
Editors:
ALAN S. MANNE
T.N. SRINIVASAN

Review Editor:
PADMA DESAI

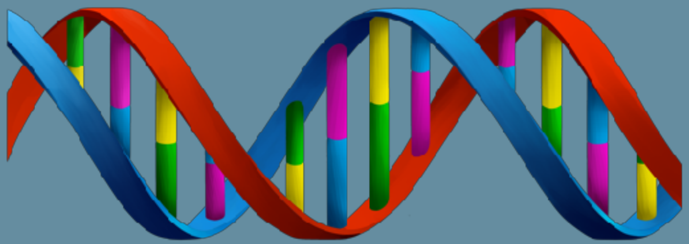
Associate Editors:
I. ADELMAN
E.L. BACHA
J.N. BHAGWATI
M. BRUNO
H.J. BRUTON
M. CELASUN
P. DASGUPTA
DIAZ ALEJANDRO
J. ENCARNACION
E. GARCIA
M. GARGOURI
D. GHAI
R. JOLLY
J.M. KATZ
L.E.M. MENNES
Y. MURAKAMI
H.M.A. ONITIRI
G. PAPANDREOU
J. SANDEE
L. SOLIS
L.G. STOLERU
L.J. TAYLOR
T.E. WEISSKOPF

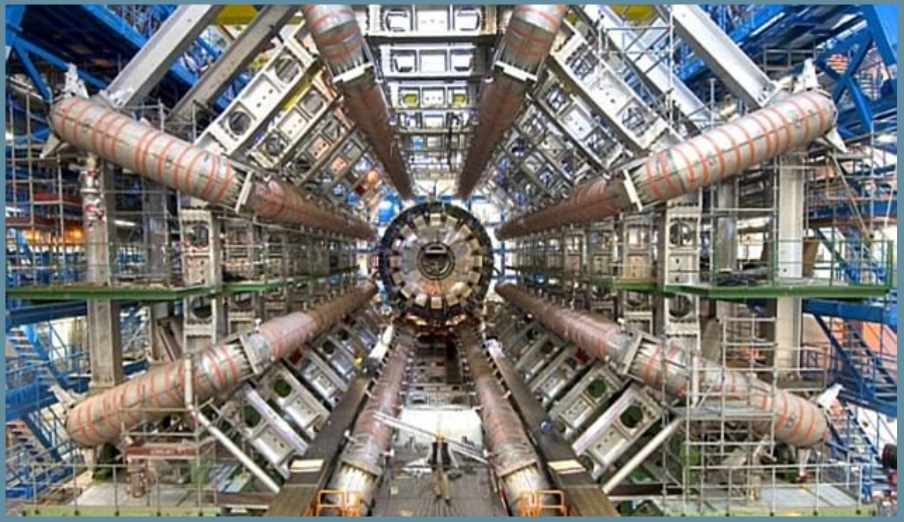
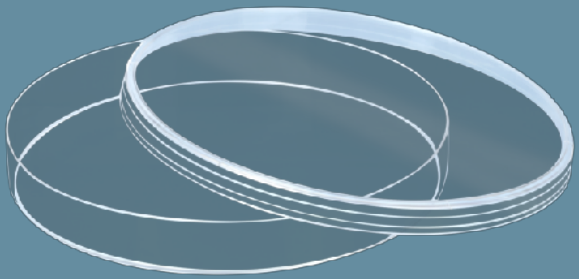
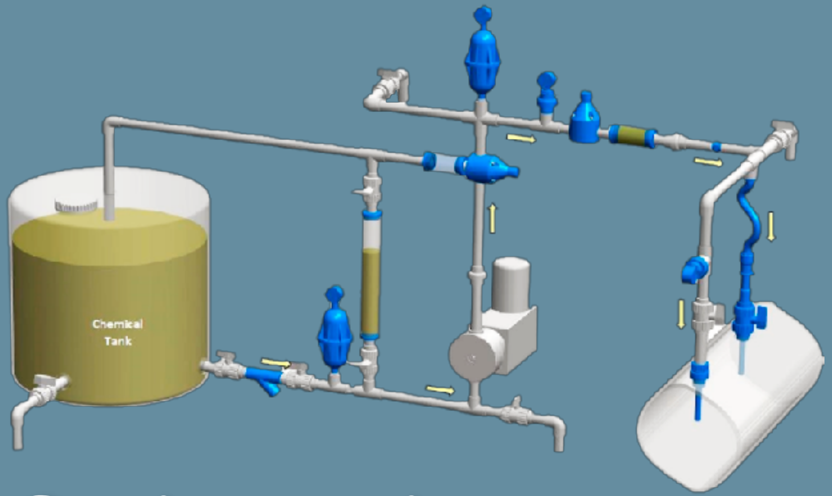
USENIX

```
\usepackage{usenix2019_v3}

%USENIX program committees
%give extra points to
%submissions that are backed
%by artifacts that are
%publicly available.

\section*{Availability}
```

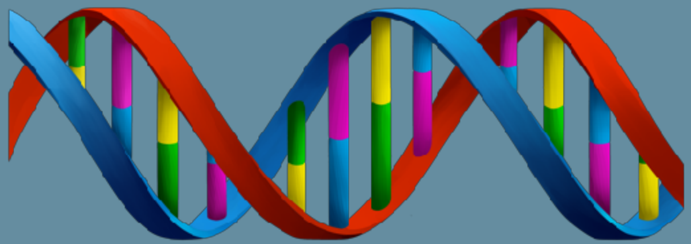




Scientist



Scientist



101010101
010100100
000111111
100100100

Computer
Scientist

101010101

nature

Authors must make available … any … computer code … used to generate results that are reported in the paper

SC19

Appendix

Artifact Description (mandatory)

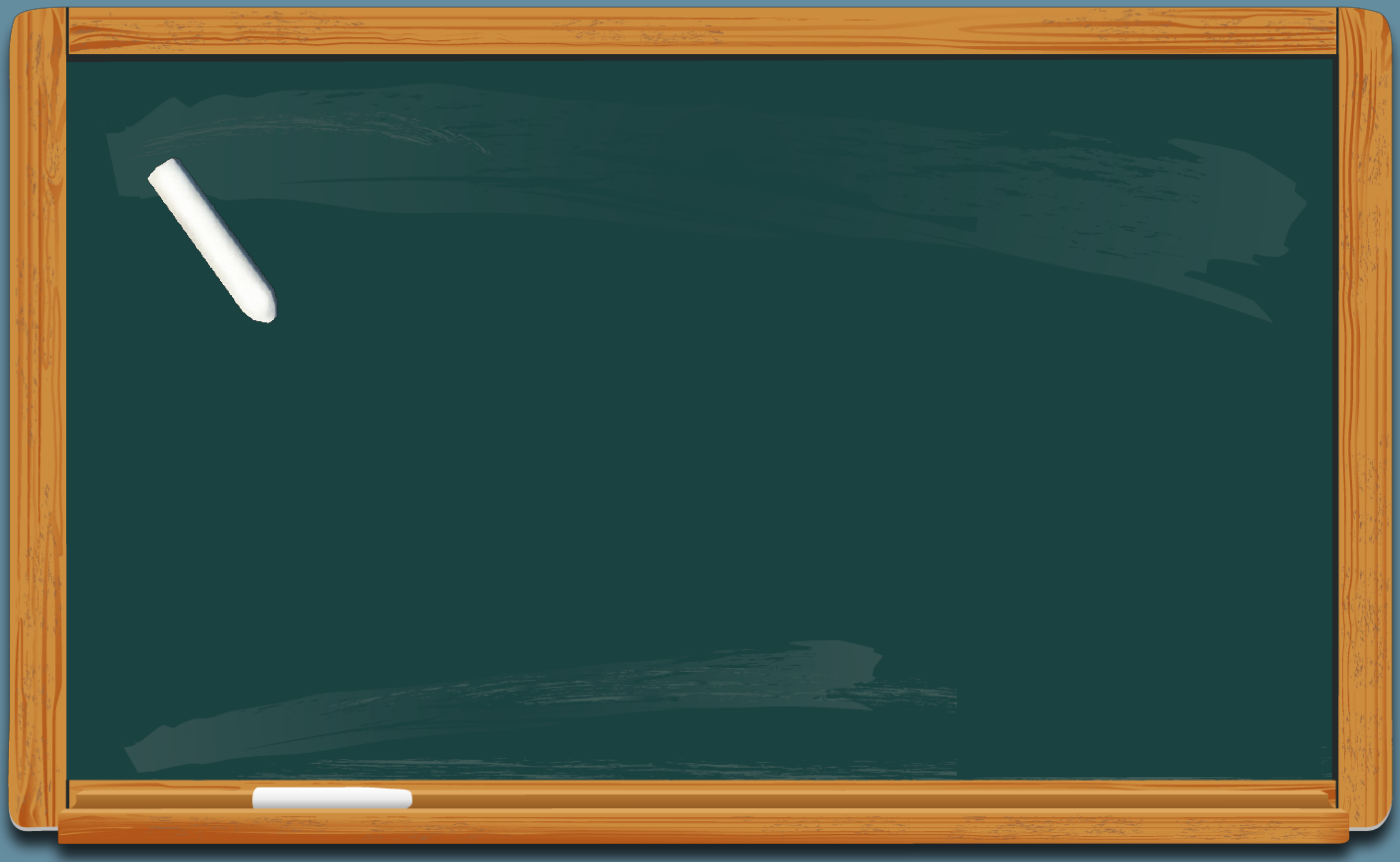
- **Summarize the experiments**
- **Artifact availability**
- **Experimental setup**

Appendix

Artifact Evaluation (optional)

- **Validate timings?**
- **Describe statistics!**

Sharing Proposal — #7 —



Sharing Proposal
— #7 —

CS Research Methods Courses?

CS Research Methods Courses?

- Reading, writing, presenting, reviewing papers
- Experimental design
- Statistics, data processing, visualization
- Proposal writing, career issues
- Intellectual property, research ethics

CS Research Methods Courses?

- Reading, writing, presenting, reviewing papers
- Experimental design
- Statistics, data processing, visualization
- Proposal writing, career issues
- Intellectual property, research ethics

Reproducibility???

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
DEPARTMENT OF MECHANICAL ENGINEERING

2.671 Measurement and Instrumentation

Keeping a complete and accurate record of experimental methods and data … **could someone else**, … use your notebook to **repeat your work**, and obtain the same results?

Reproducibility PI Manifesto

Lorena Barba

I pledge to

- ☑ teach grad students about reproducibility
- ☑ share artifacts at the time of submission
- ☑ add a *reproducibility statement* to papers

http://lorenabarba.com/gallery/reproducibility-pi-manifesto/

The dissertation proposal should state **if and how** they will provide access to code and data to support reproducibility.

Sharing Proposal #8

— #8 —

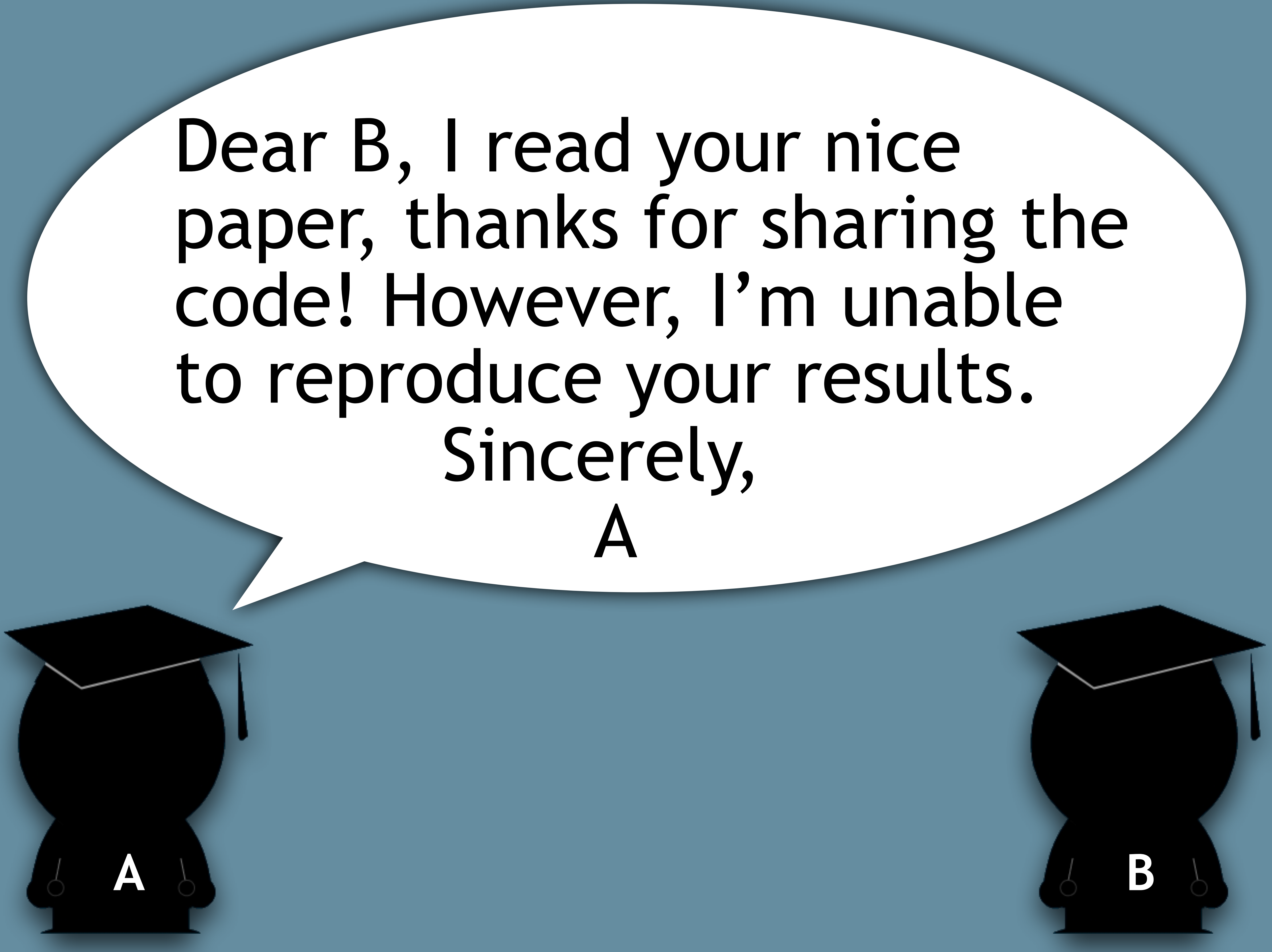
All I Really Need to Know I Learned in Kindergarten

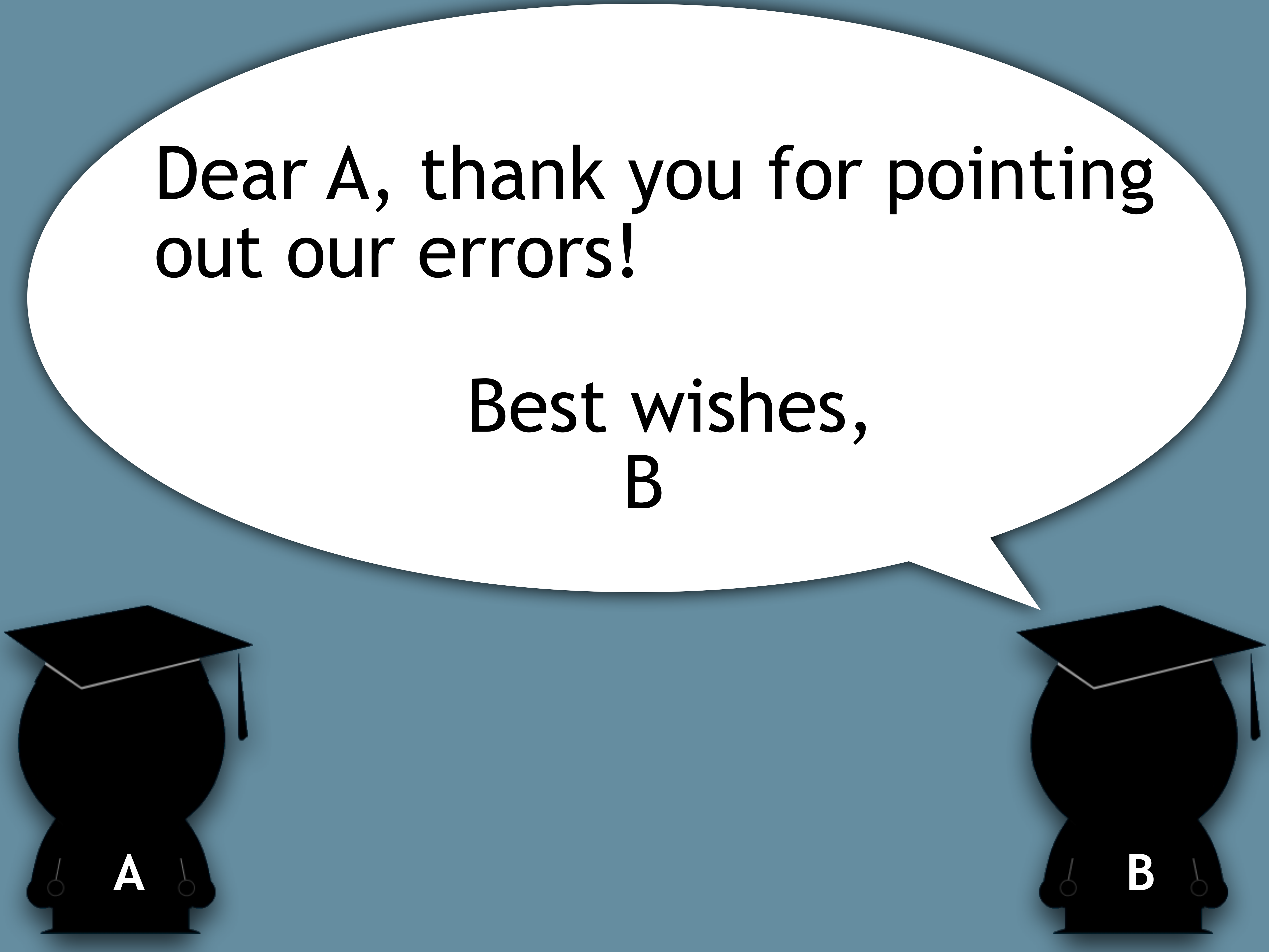
Why do we care about reproducibility and repeatability?

Why do we care about reproducibility and repeatability?

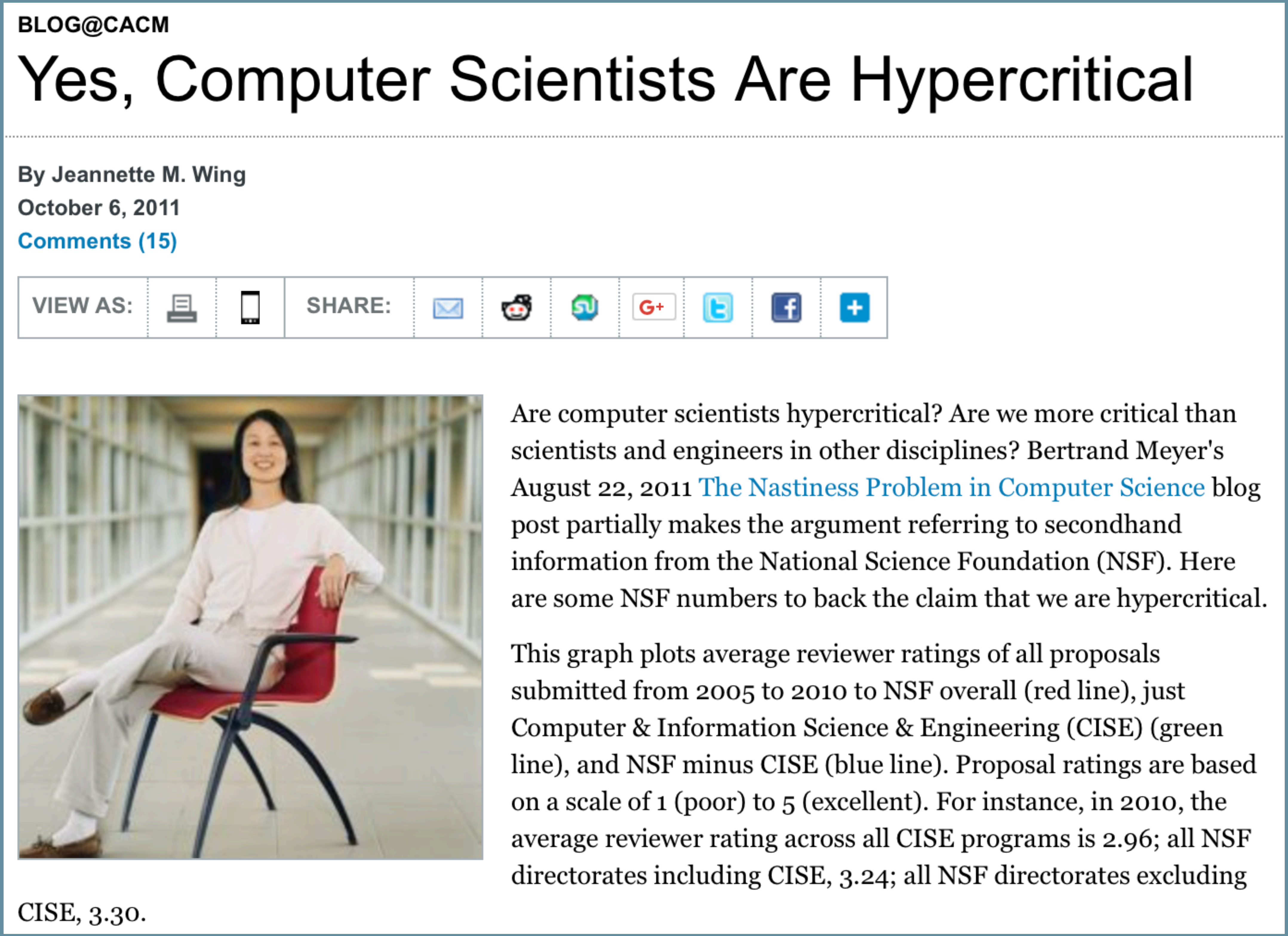
Why do we care about reproducibility and repeatability?

Why do we care about reproducibility and repeatability?


Dear B, I read your nice paper, thanks for sharing the code! However, I'm unable to reproduce your results.
Sincerely,
A
A
B


Dear A, thank you for pointing out our errors!
Best wishes,
B
A
B

Dear A, thank you for pointing out our errors!
Best wishes,
B
Respect
A



BLOG@CACM

Yes, Computer Scientists Are Hypercritical

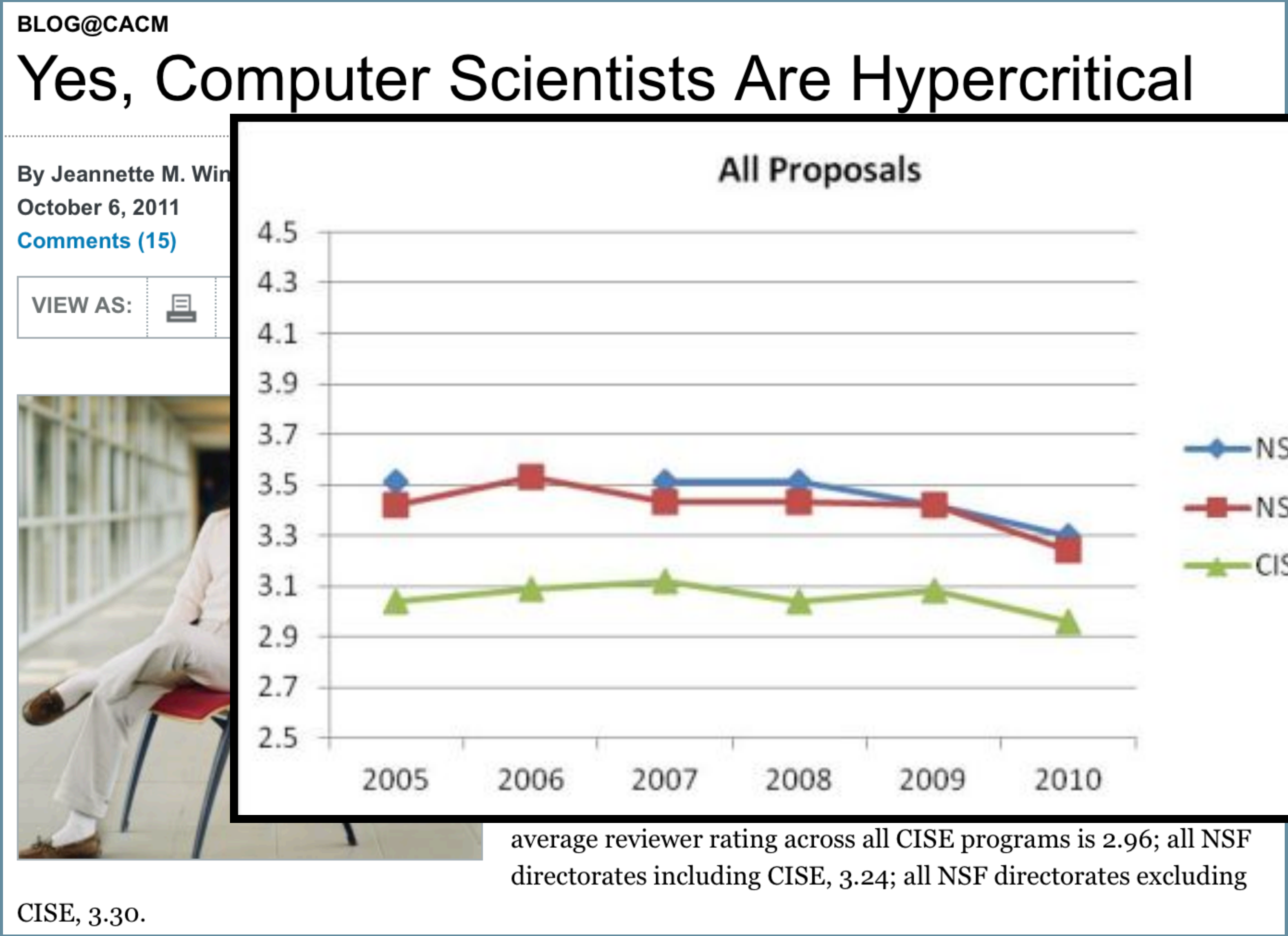
By Jeannette M. Wing
October 6, 2011
Comments (15)

VIEW AS: SHARE:

Are computer scientists hypercritical? Are we more critical than scientists and engineers in other disciplines? Bertrand Meyer's August 22, 2011 The Nastiness Problem in Computer Science blog post partially makes the argument referring to secondhand information from the National Science Foundation (NSF). Here are some NSF numbers to back the claim that we are hypercritical.

This graph plots average reviewer ratings of all proposals submitted from 2005 to 2010 to NSF overall (red line), just Computer & Information Science & Engineering (CISE) (green line), and NSF minus CISE (blue line). Proposal ratings are based on a scale of 1 (poor) to 5 (excellent). For instance, in 2010, the average reviewer rating across all CISE programs is 2.96; all NSF directorates including CISE, 3.24; all NSF directorates excluding CISE, 3.30.

https://cacm.acm.org/blogs/blog-cacm/134743-yes-computer-scientists-are-hypercritical/fulltext



BLOG@CACM

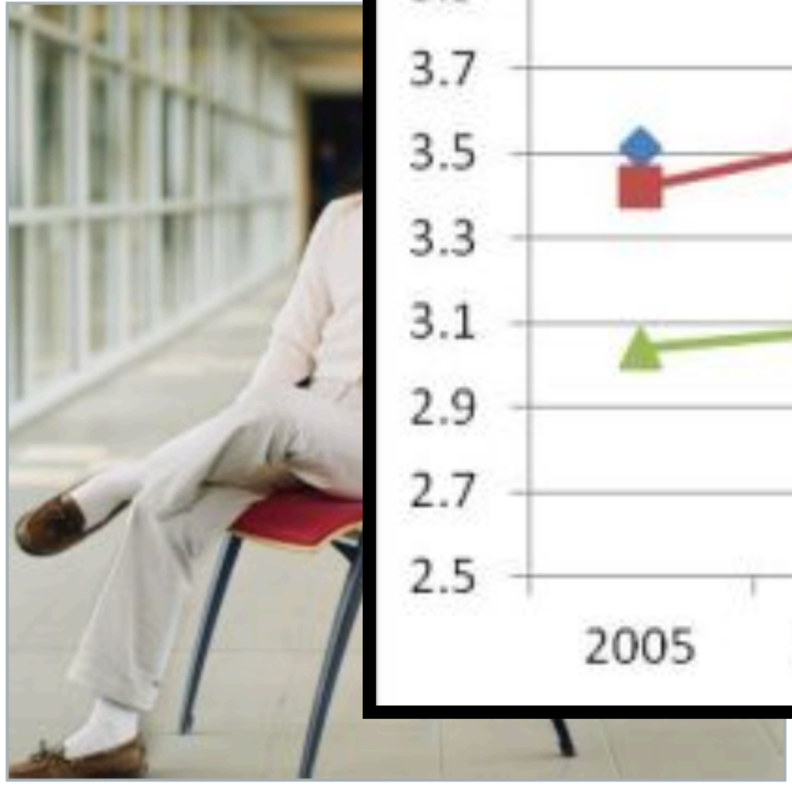
Yes, Computer Scientists Are Hypercritical

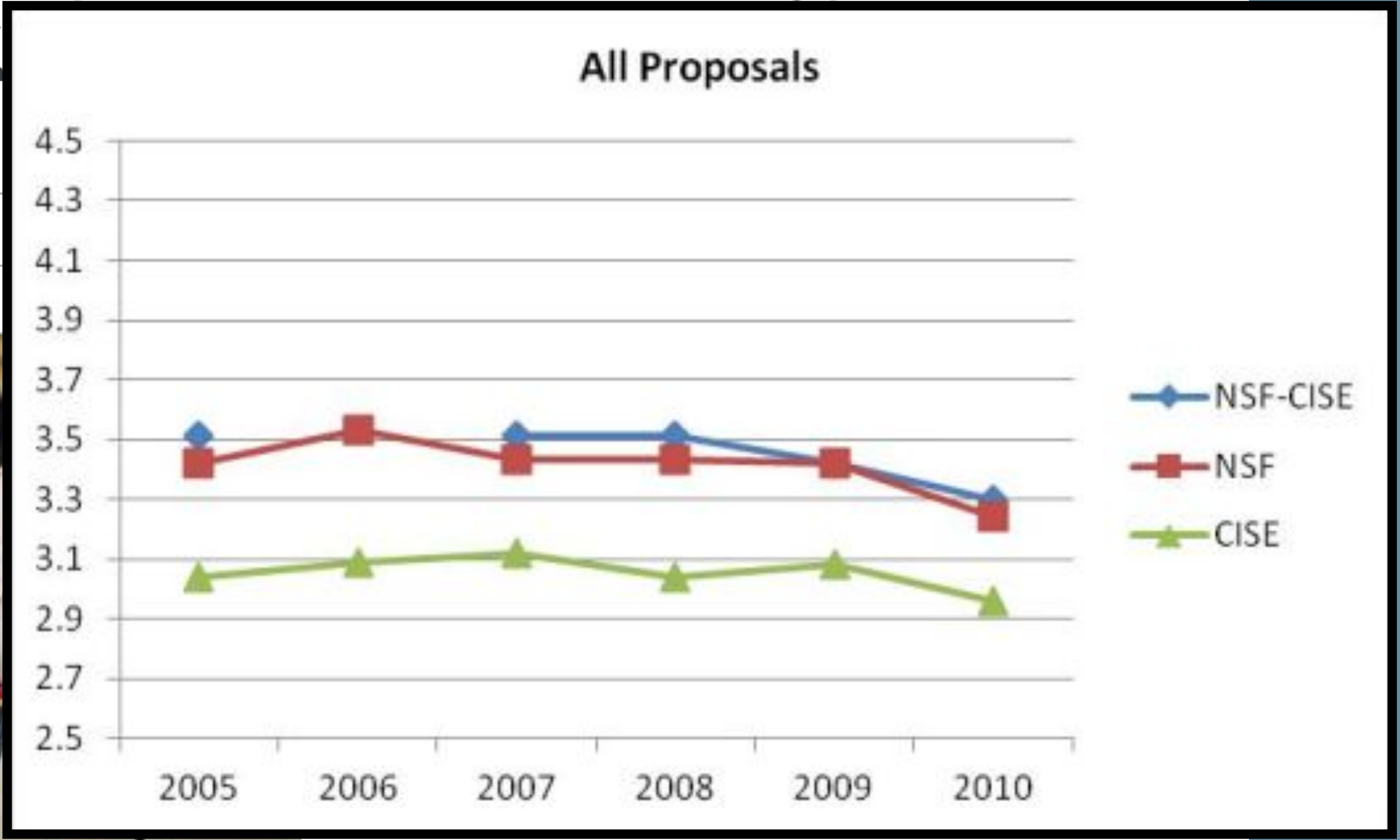
By Jeannette M. Win

October 6, 2011

Comments (15)

VIEW AS:





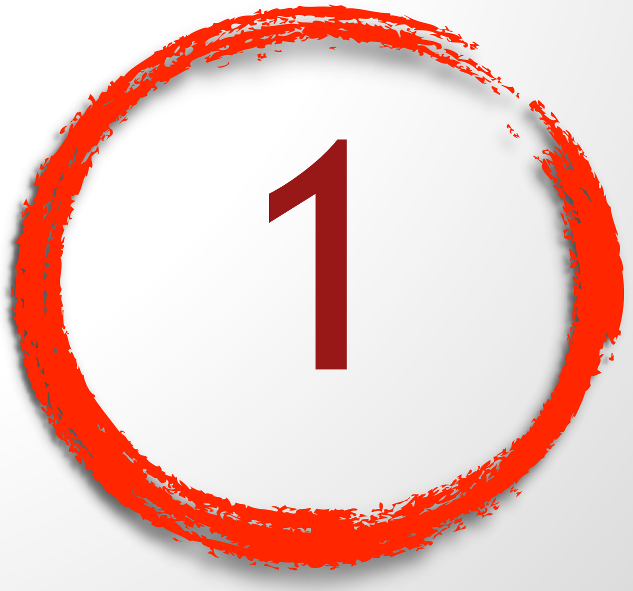
average reviewer rating across all CISE programs is 2.96; all NSF directorates including CISE, 3.24; all NSF directorates excluding CISE, 3.30.

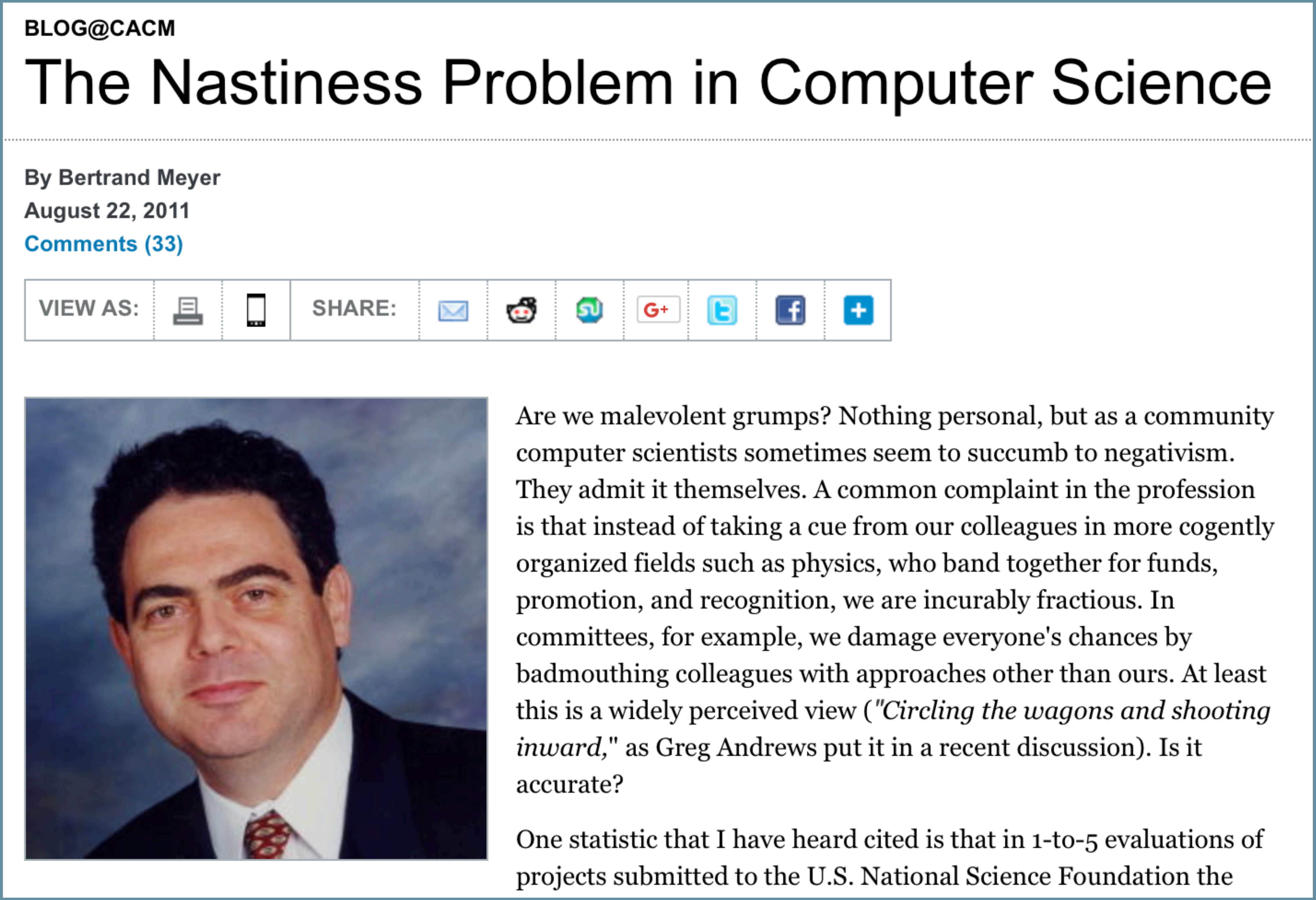
https://cacm.acm.org/blogs/blog-cacm/134743-yes-computer-scientists-are-hypercritical/fulltext

DBMS Research First 50 Years, Next 50 Years

Jeffrey F. Naughton

- SIGMOD 2010
- 350 submissions
- Number of papers with all reviews "accept" or higher:





BLOG@CACM

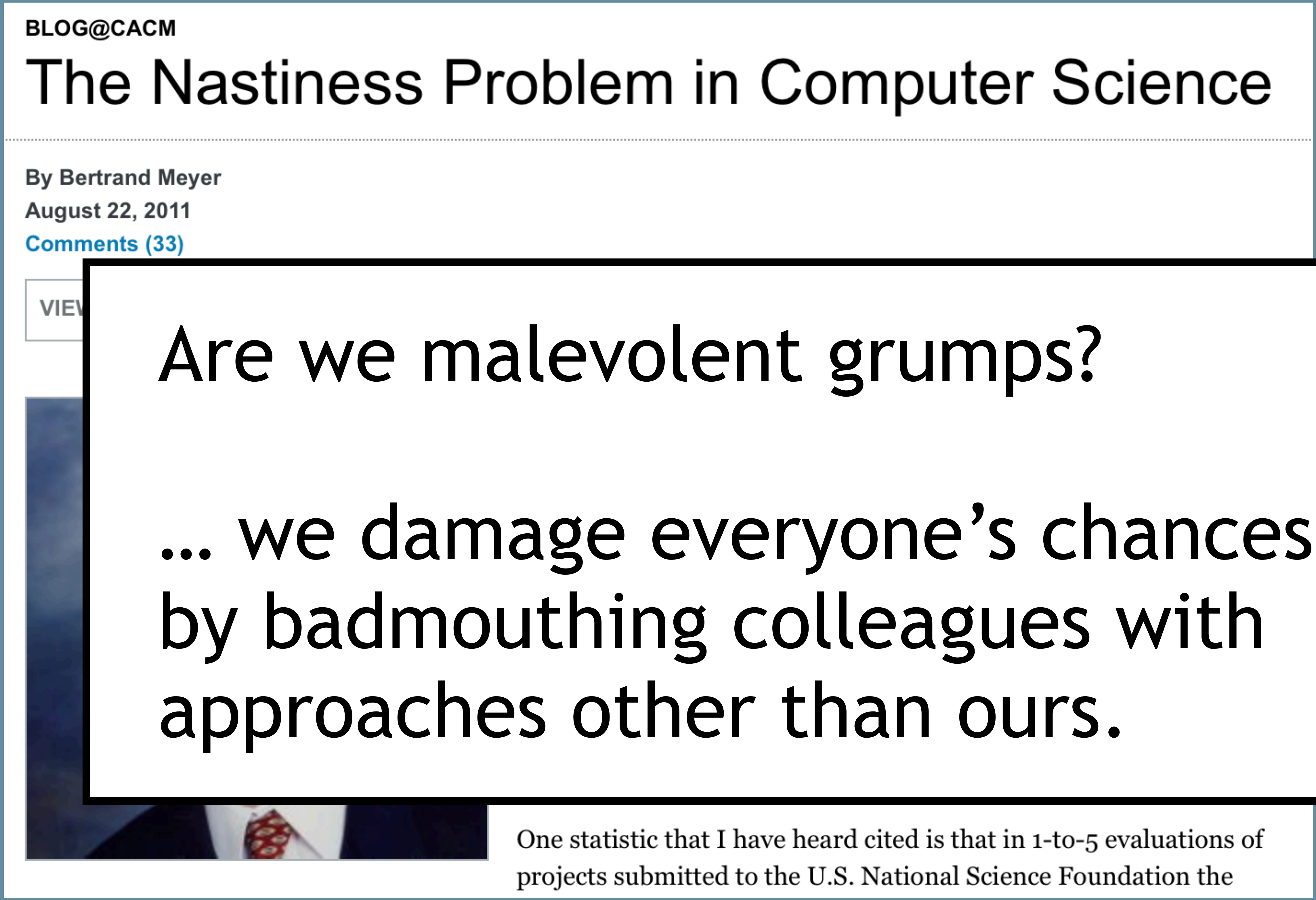
The Nastiness Problem in Computer Science

By Bertrand Meyer
August 22, 2011
Comments (33)

Are we malevolent grumps? Nothing personal, but as a community computer scientists sometimes seem to succumb to negativism. They admit it themselves. A common complaint in the profession is that instead of taking a cue from our colleagues in more cogently organized fields such as physics, who band together for funds, promotion, and recognition, we are incurably fractious. In committees, for example, we damage everyone's chances by badmouthing colleagues with approaches other than ours. At least this is a widely perceived view (*"Circling the wagons and shooting inward,"* as Greg Andrews put it in a recent discussion). Is it accurate?

One statistic that I have heard cited is that in 1-to-5 evaluations of projects submitted to the U.S. National Science Foundation the

https://cacm.acm.org/blogs/blog-cacm/123611-the-nastiness-problem-in-computer-science/fulltext



BLOG@CACM

The Nastiness Problem in Computer Science

By Bertrand Meyer
August 22, 2011
Comments (33)

VIE

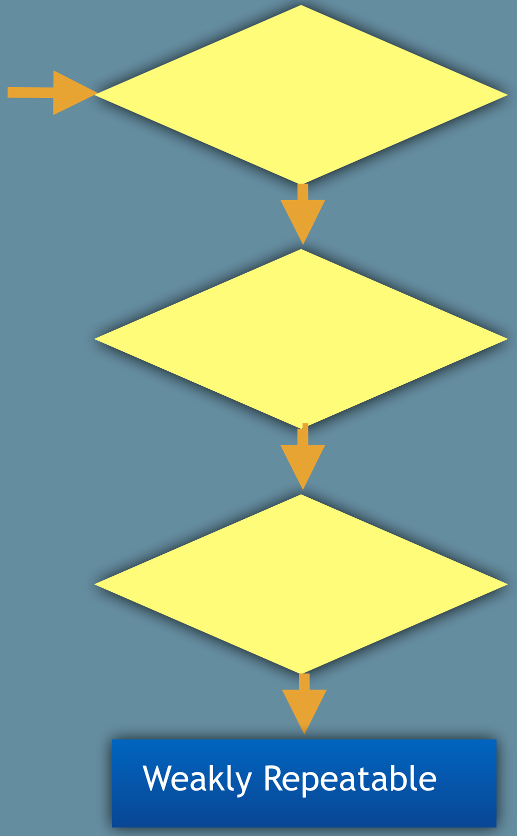
Are we malevolent grumps?

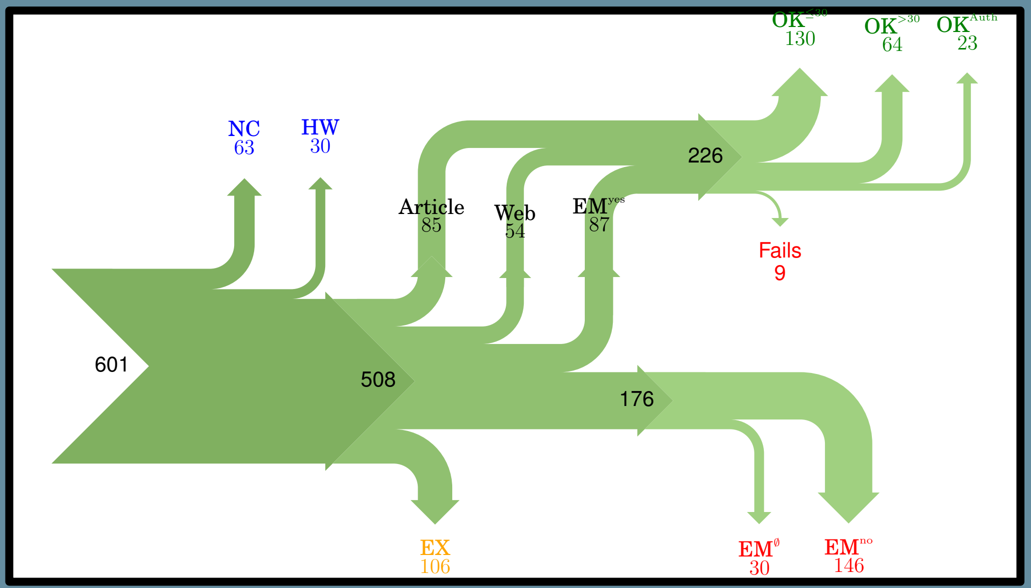
… we damage everyone's chances by badmouthing colleagues with approaches other than ours.

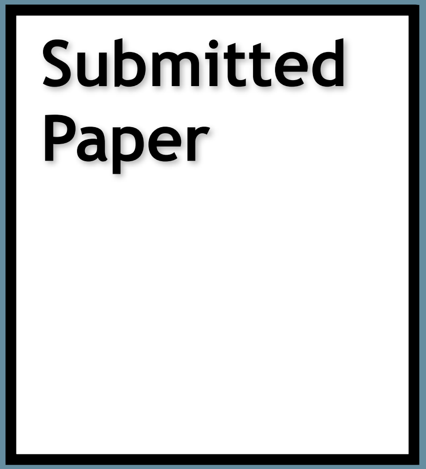
One statistic that I have heard cited is that in 1-to-5 evaluations of projects submitted to the U.S. National Science Foundation the

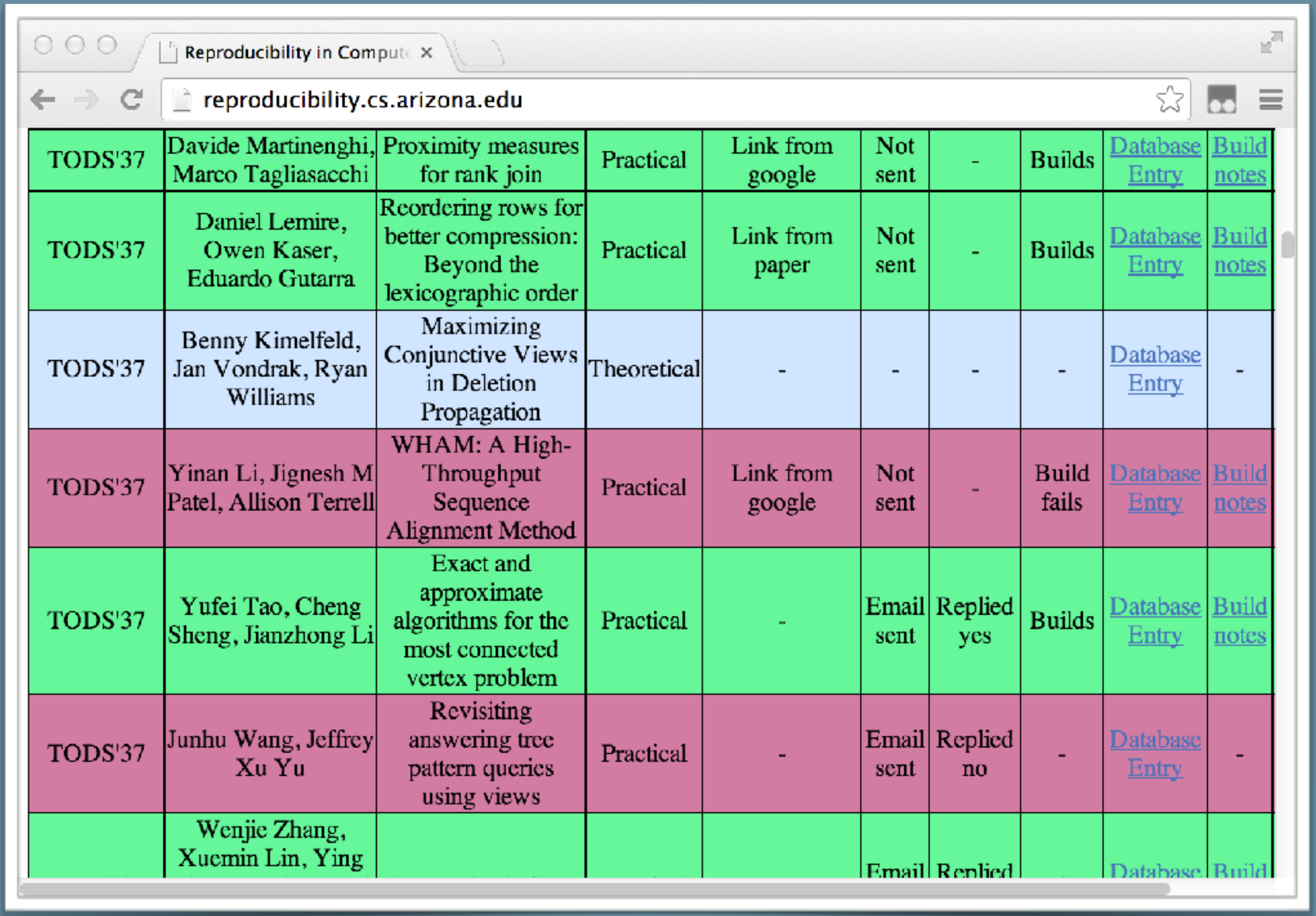
https://cacm.acm.org/blogs/blog-cacm/123611-the-nastiness-problem-in-computer-science/fulltext

What Happened Next?

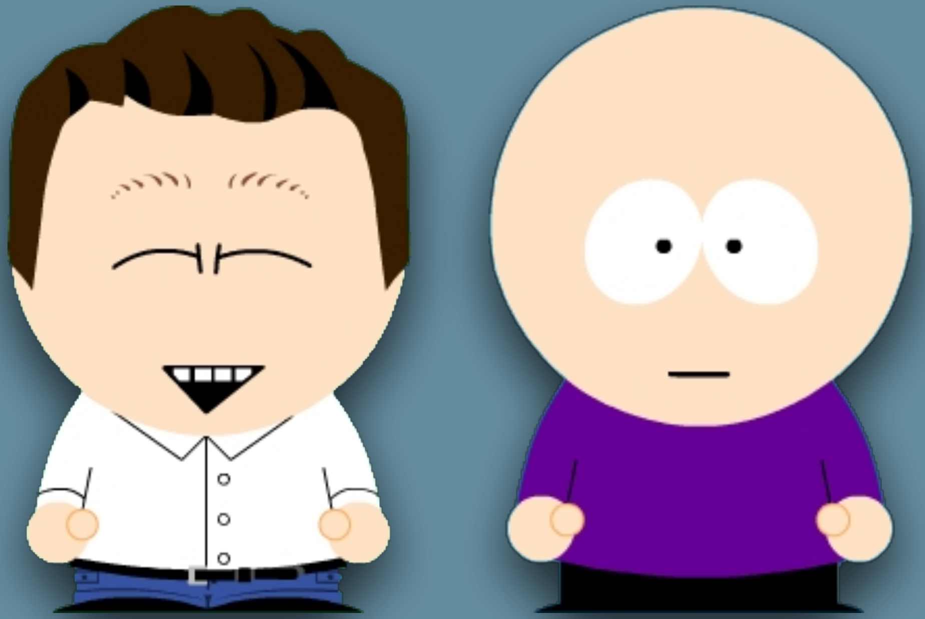




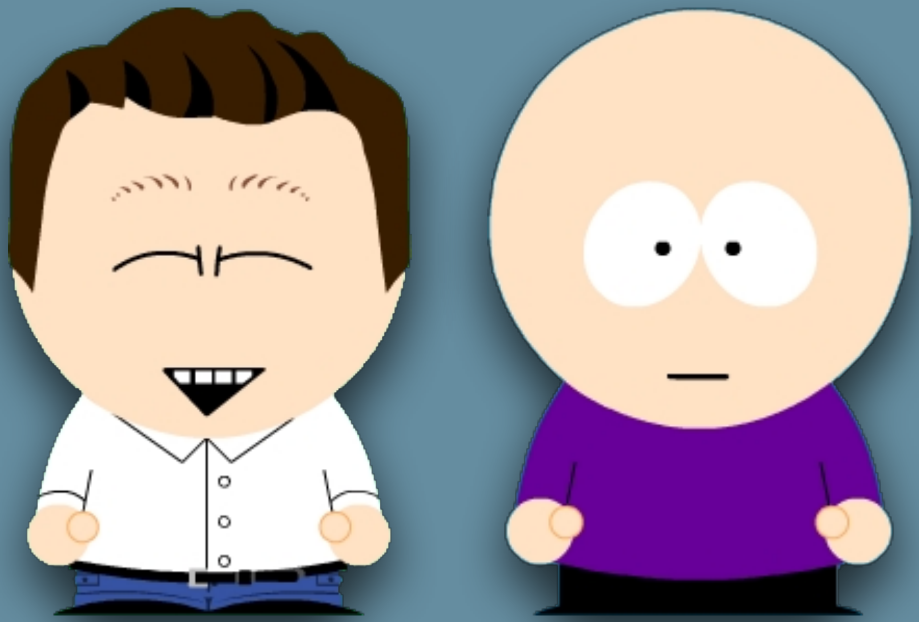




TODS'37	Davide Martinenghi, Marco Tagliasacchi	Proximity measures for rank join	Practical	Link from google	Not sent	-	Builds	Database Entry	Build notes
TODS'37	Daniel Lemire, Owen Kaser, Eduardo Gutarra	Reordering rows for better compression: Beyond the lexicographic order	Practical	Link from paper	Not 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

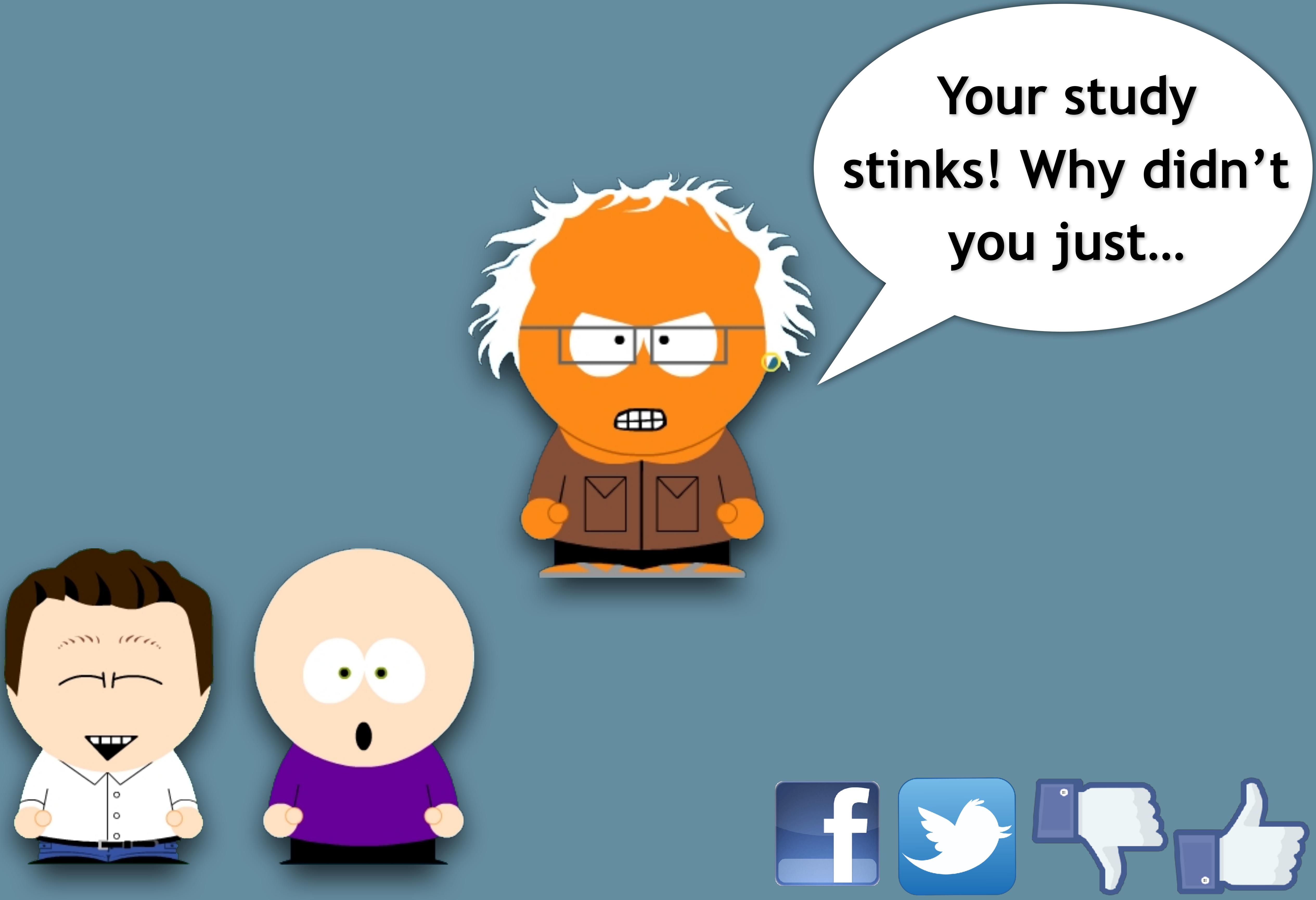


Hate Us on Facebook!

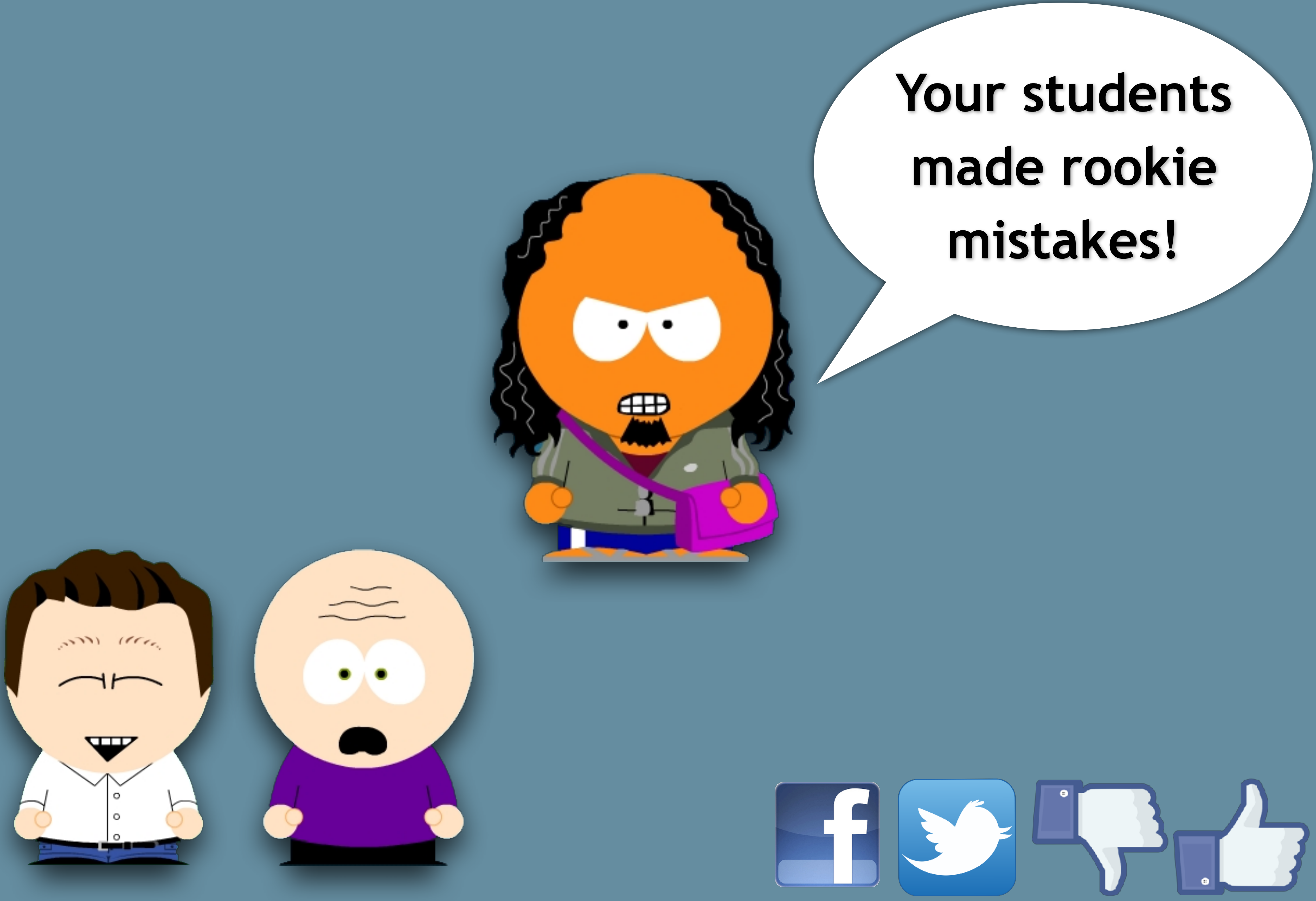


Hate Us on Facebook!

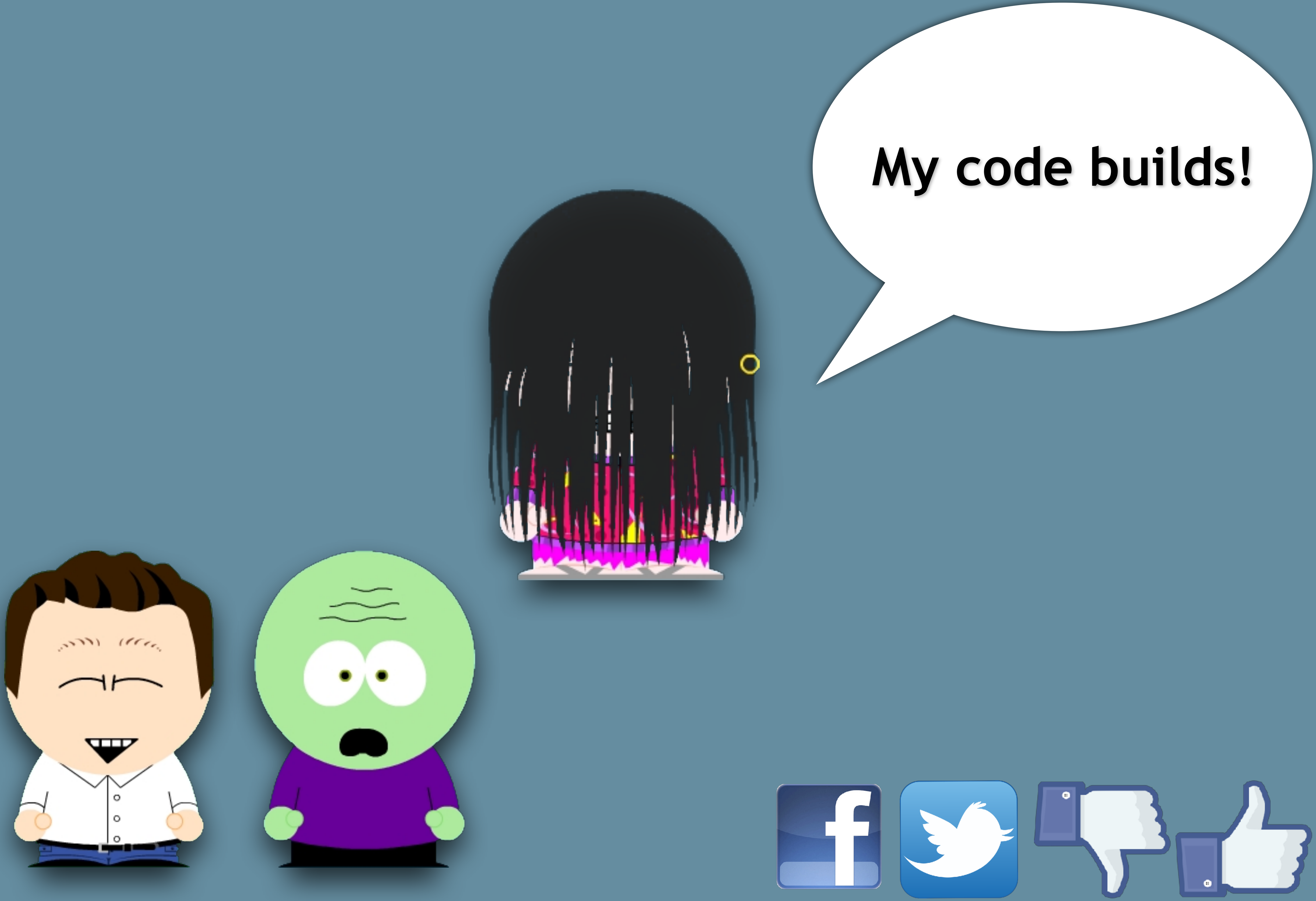
Hate Us on Facebook!



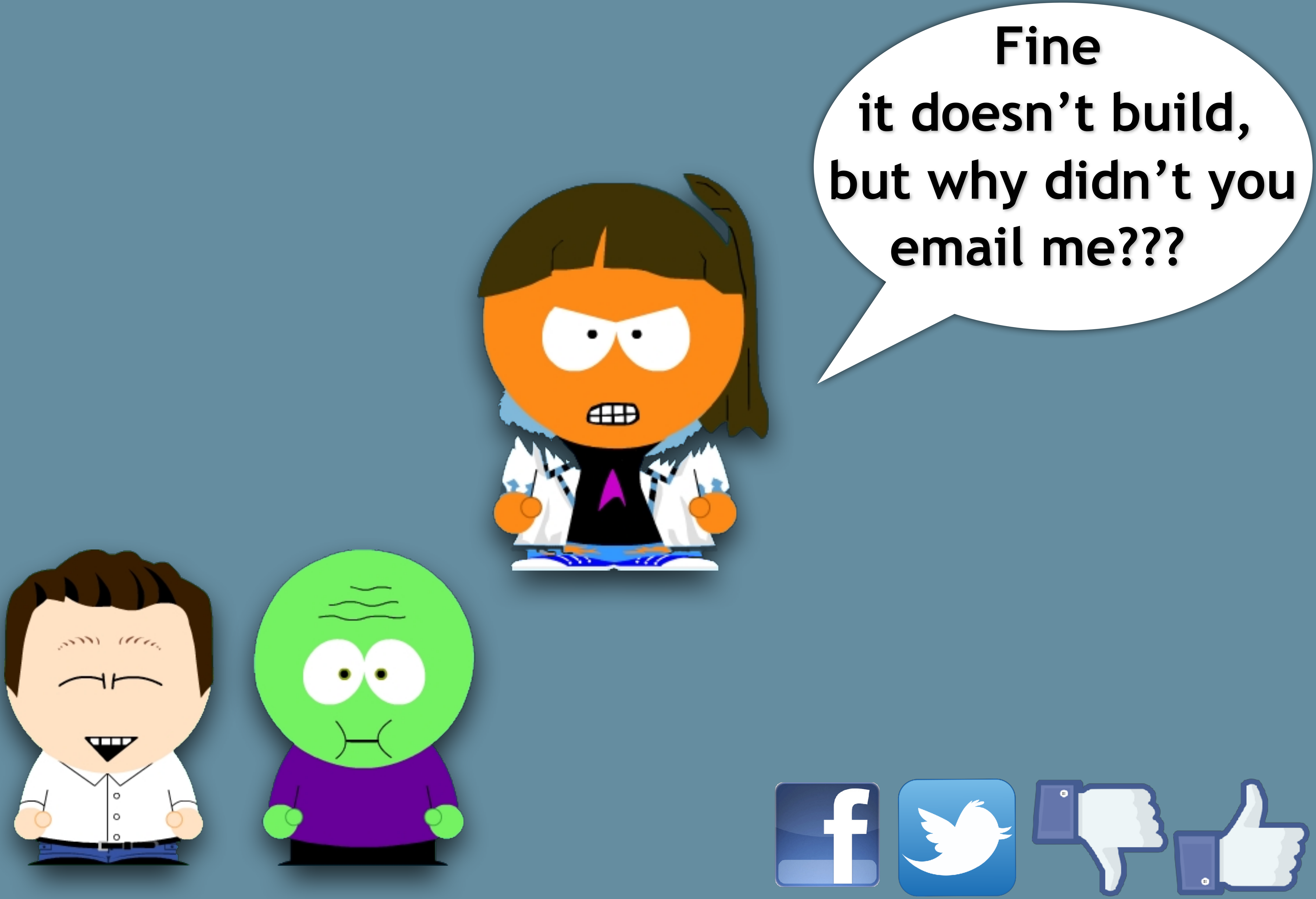
Hate Us on Facebook!



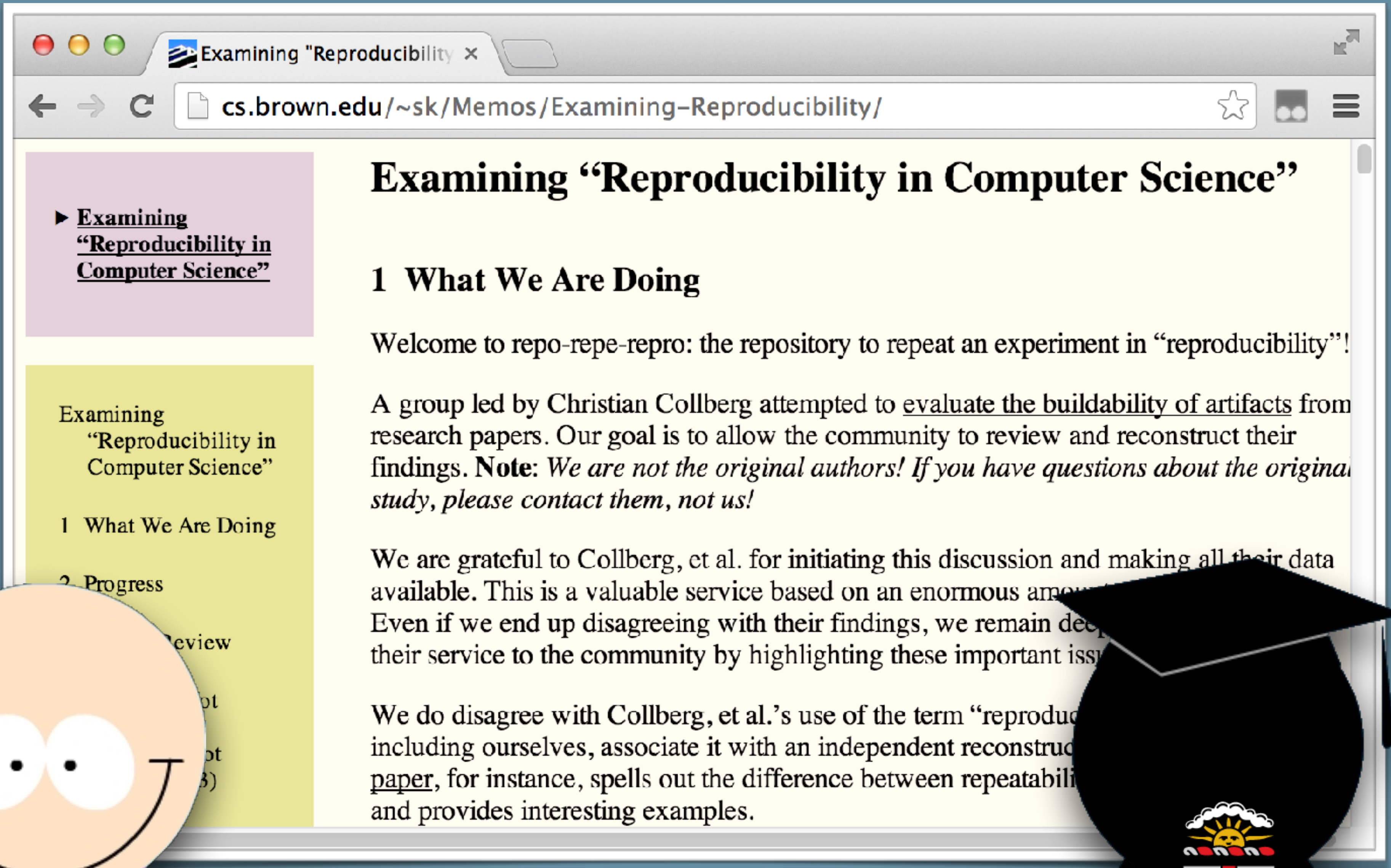
Hate Us on Facebook!



Hate Us on Facebook!



Turnabout is Fair Play!



Examining "Reproducibility

cs.brown.edu/~sk/Memos/Examining-Reproducibility/

▸ Examining "Reproducibility in Computer Science"

Examining "Reproducibility in Computer Science"

1 What We Are Doing

2 Progress

Examining "Reproducibility in Computer Science"

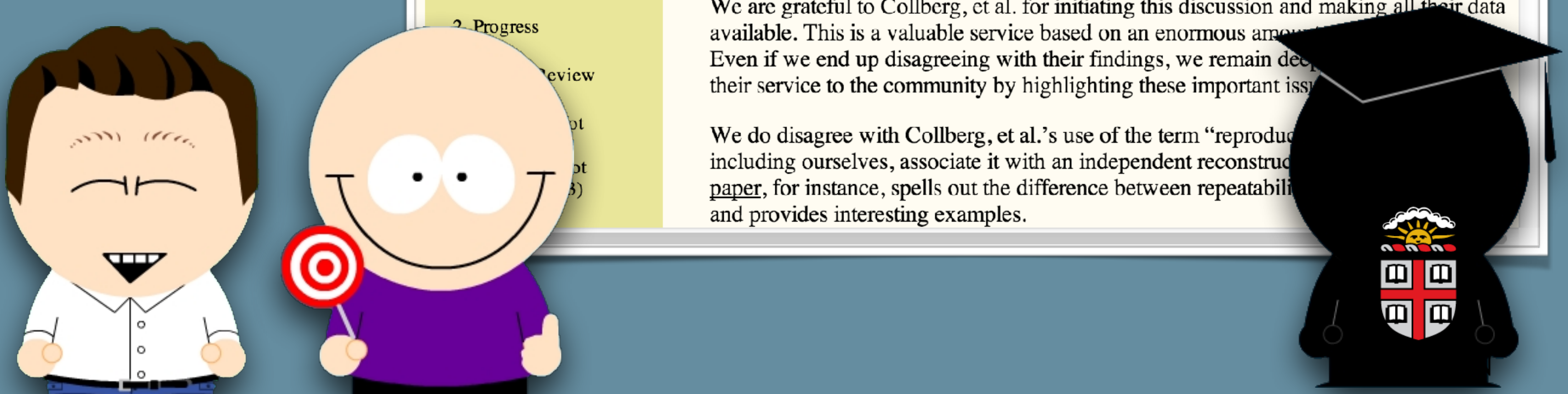
1 What We Are Doing

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

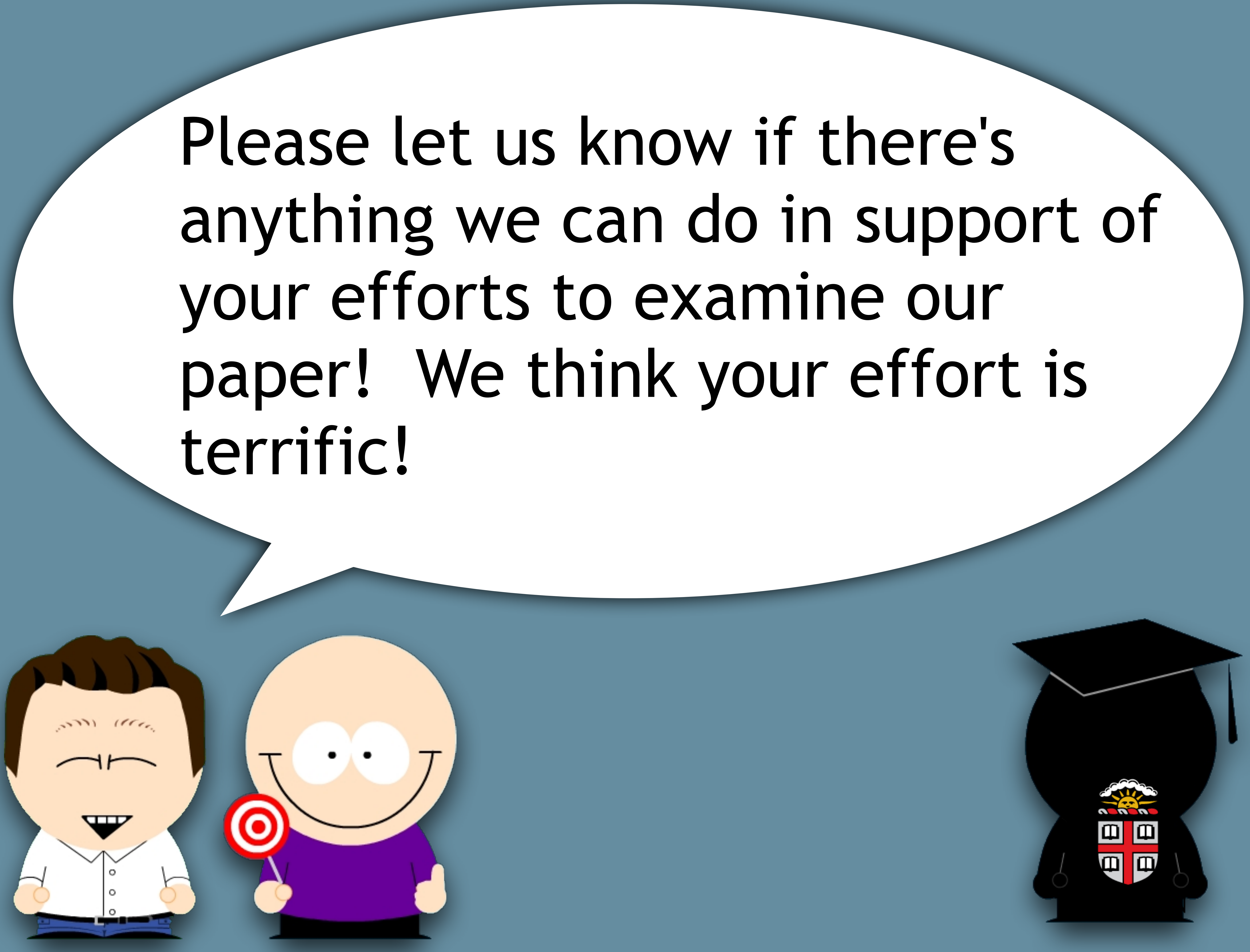
A group led by Christian Collberg attempted to evaluate the buildability of artifacts from research papers. Our goal is to allow the community to review and reconstruct their findings. **Note**: *We are not the original authors! If you have questions about the original study, please contact them, not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

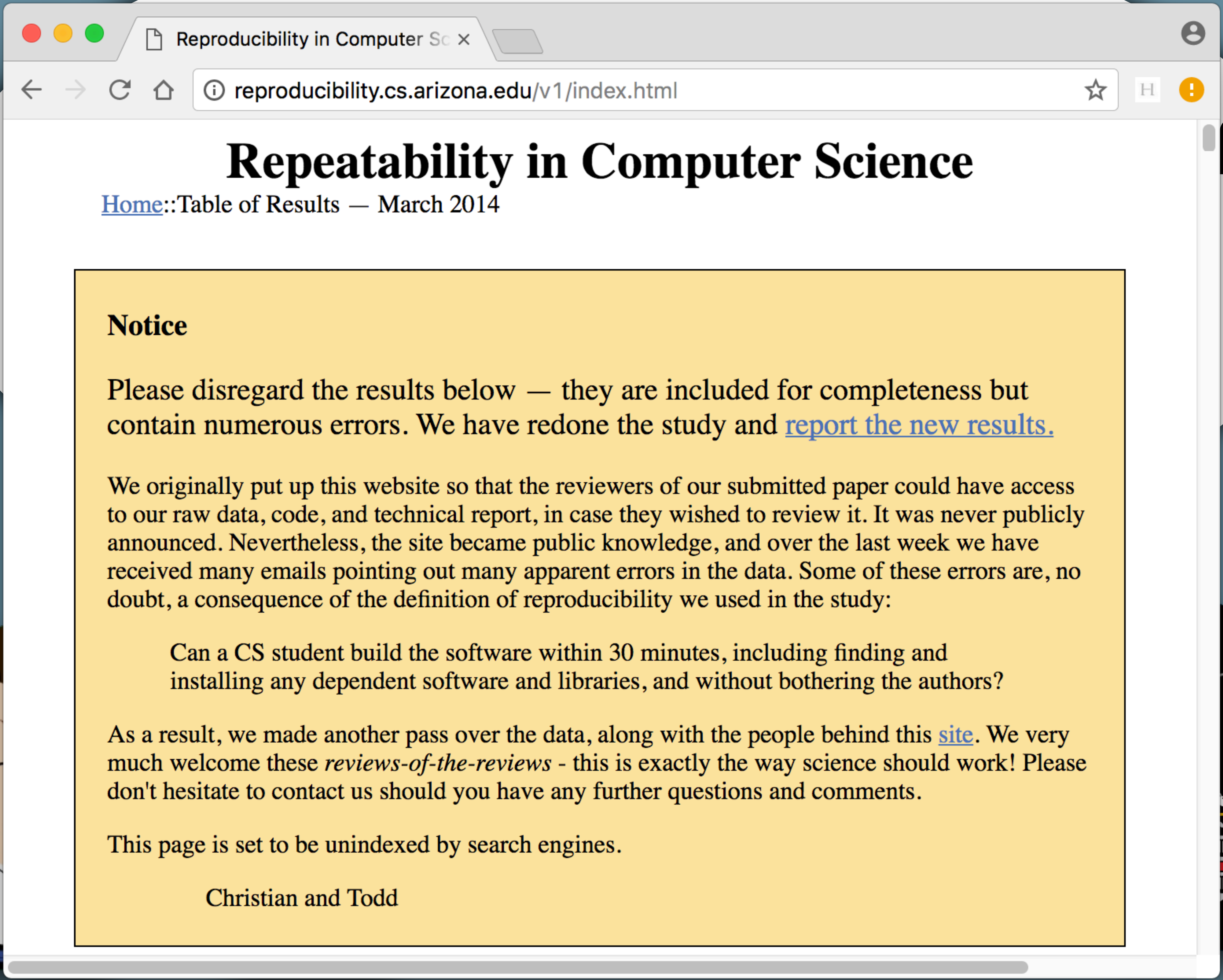
We are grateful to Collberg, et al. for initiating this discussion and making all their data available. This is a valuable service based on an enormous am
Even if we end up disagreeing with their findings, we remain de
their service to the community by highlighting these important iss

We do disagree with Collberg, et al.'s use of the term "reproduc
including ourselves, associate it with an independent reconstru
paper, for instance, spells out the difference between repeatabili
and provides interesting examples.



http://cs.brown.edu/~sk/Memos/Examining-Reproducibility/


Please let us know if there's anything we can do in support of your efforts to examine our paper! We think your effort is terrific!


Reproducibility in Computer Sc
reproducibility.cs.arizona.edu/v1/index.html

Repeatability in Computer Science

Home::Table of Results — March 2014

Notice

Please disregard the results below — they are included for completeness but contain numerous errors. We have redone the study and report the new results.

We originally put up this website so that the reviewers of our submitted paper could have access to our raw data, code, and technical report, in case they wished to review it. It was never publicly announced. Nevertheless, the site became public knowledge, and over the last week we have received many emails pointing out many apparent errors in the data. Some of these errors are, no doubt, a consequence of the definition of reproducibility we used in the study:

> Can a CS student build the software within 30 minutes, including finding and installing any dependent software and libraries, and without bothering the authors?

As a result, we made another pass over the data, along with the people behind this site. We very much welcome these *reviews-of-the-reviews* - this is exactly the way science should work! Please don't hesitate to contact us should you have any further questions and comments.

This page is set to be unindexed by search engines.

Christian and Todd

contributed articles

To encourage repeatable re
repeatability engineering a
commitments to sharing re

BY CHRISTIAN COLLBERG AND TODD

Repeatability in Computer Systems Research

IN 2012, WHEN reading a paper f
computer security conference, w
there is a clever way to defeat th
in the paper, and, in order to sh
the authors (faculty and graduat
ranked U.S. computer science d
for access to their prototype syst
no response. We thus decided to
algorithms in the paper but soo
obstacles, including a variable u
function defined but never used
formula that did not typecheck. We asked the authors
for clarification and received a single response: "I
unfortunately have few recollections of the work ... "

We next made a formal request to the university for
the source code under the broad Open Records Act
(ORA) of the authors' home state. The university's

backed up, we made a second ORA request, this time for the email messages among the authors, hoping to trace the whereabouts of the source code. The legal department first responded with: "... the records will not be produced pursuant to [ORA sub-clause]." When we pointed out reasons why this clause does not apply, the university relented but demanded $2,263.66 " ... to search for, retrieve, redact and produce such records." We declined the offer.

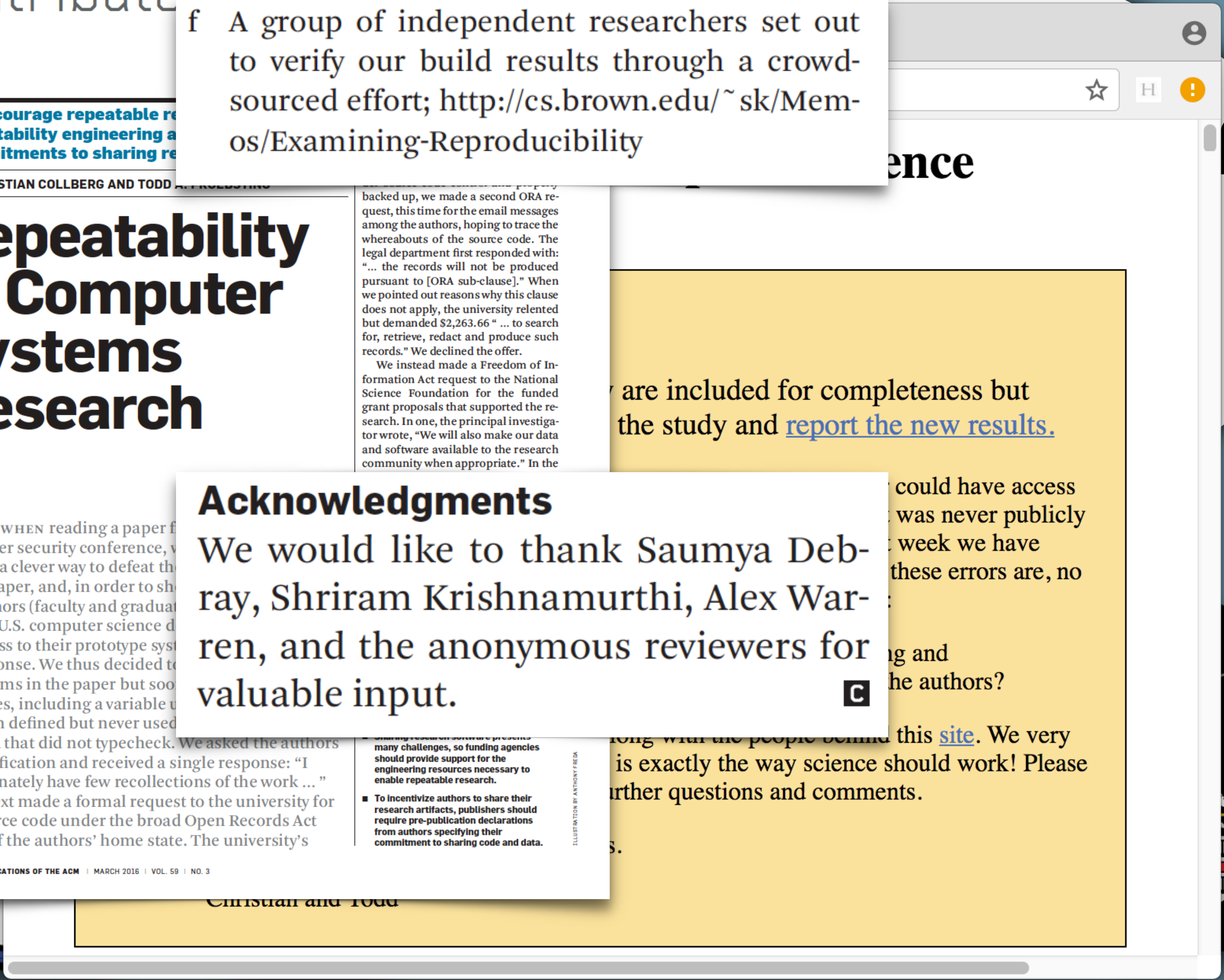
We instead made a Freedom of Information Act request to the National Science Foundation for the funded grant proposals that supported the research. In one, the principal investigator wrote, "We will also make our data and software available to the research community when appropriate." In the

many challenges, so funding agencies should provide support for the engineering resources necessary to enable repeatable research.

- To incentivize authors to share their research artifacts, publishers should require pre-publication declarations from authors specifying their commitment to sharing code and data.

ILLUSTRATION BY ANTHONY FREDA

f A group of independent researchers set out to verify our build results through a crowd-sourced effort; http://cs.brown.edu/˜sk/Memos/Examining-Reproducibility



ence

are included for completeness but
the study and report the new results.

could have access
was never publicly
week we have
these errors are, no

g and
he authors?

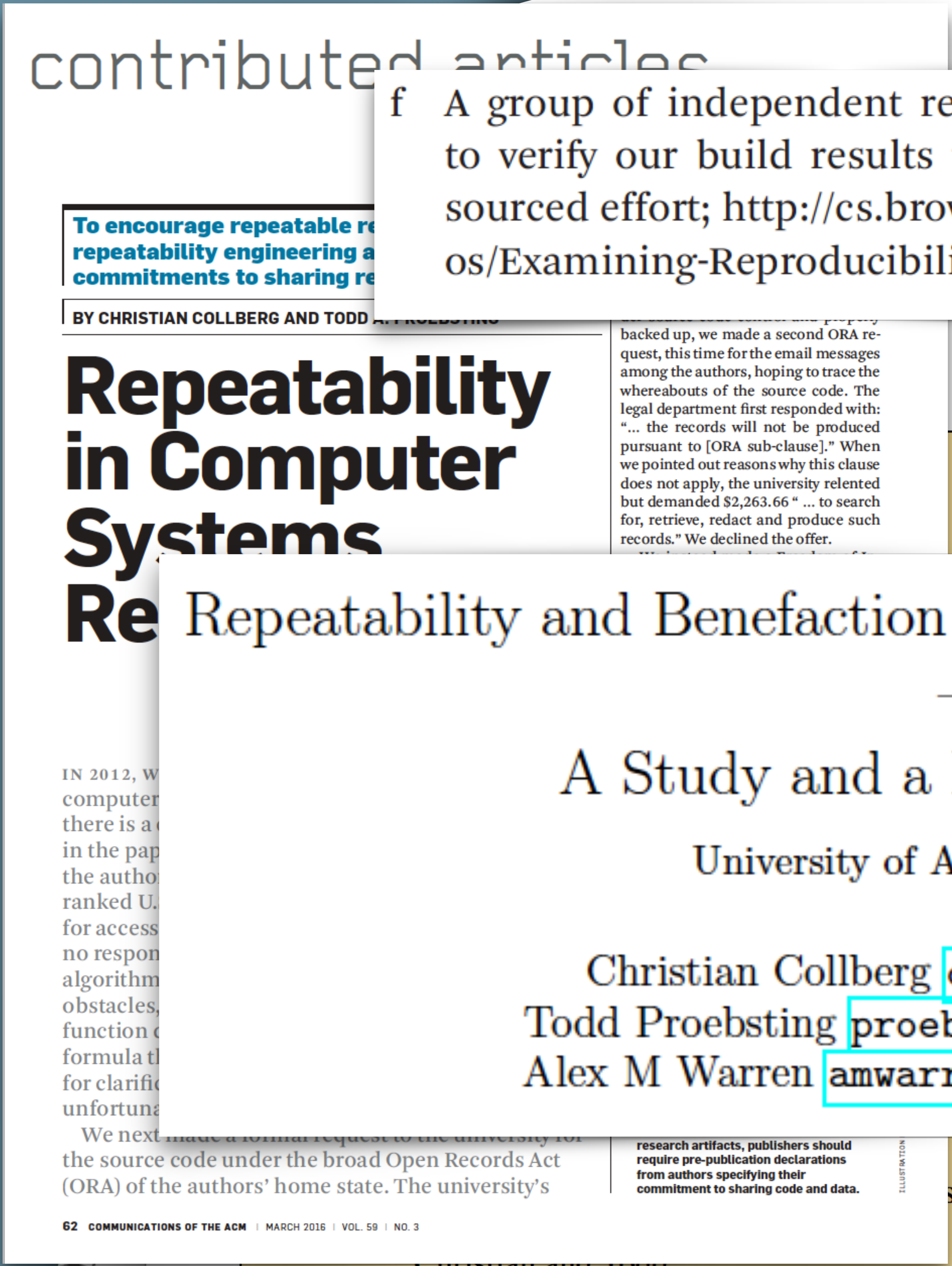
this site. We very
is exactly the way science should work! Please
rther questions and comments.

s.

Christian and Todd

Acknowledgments

We would like to thank Saumya Debray, Shriram Krishnamurthi, Alex Warren, and the anonymous reviewers for valuable input.



contributed articles

To encourage repeatable re
repeatability engineering a
commitments to sharing re

BY CHRISTIAN COLLBERG AND TODD

Repeatability in Computer Systems Re

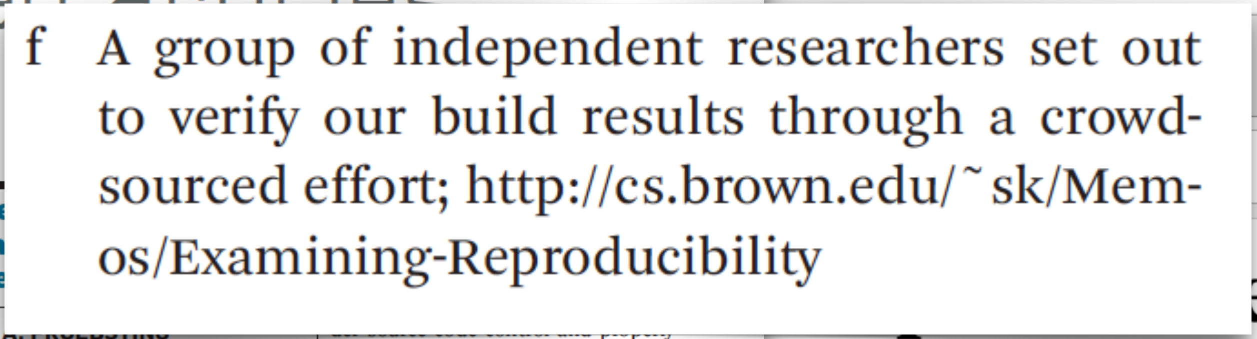
IN 2012, W
computer
there is a
in the pap
the autho
ranked U.
for access
no respon
algorithm
obstacles,
function
formula t
for clarifi
unfortuna
We next

the source code under the broad Open Records Act (ORA) of the authors' home state. The university's

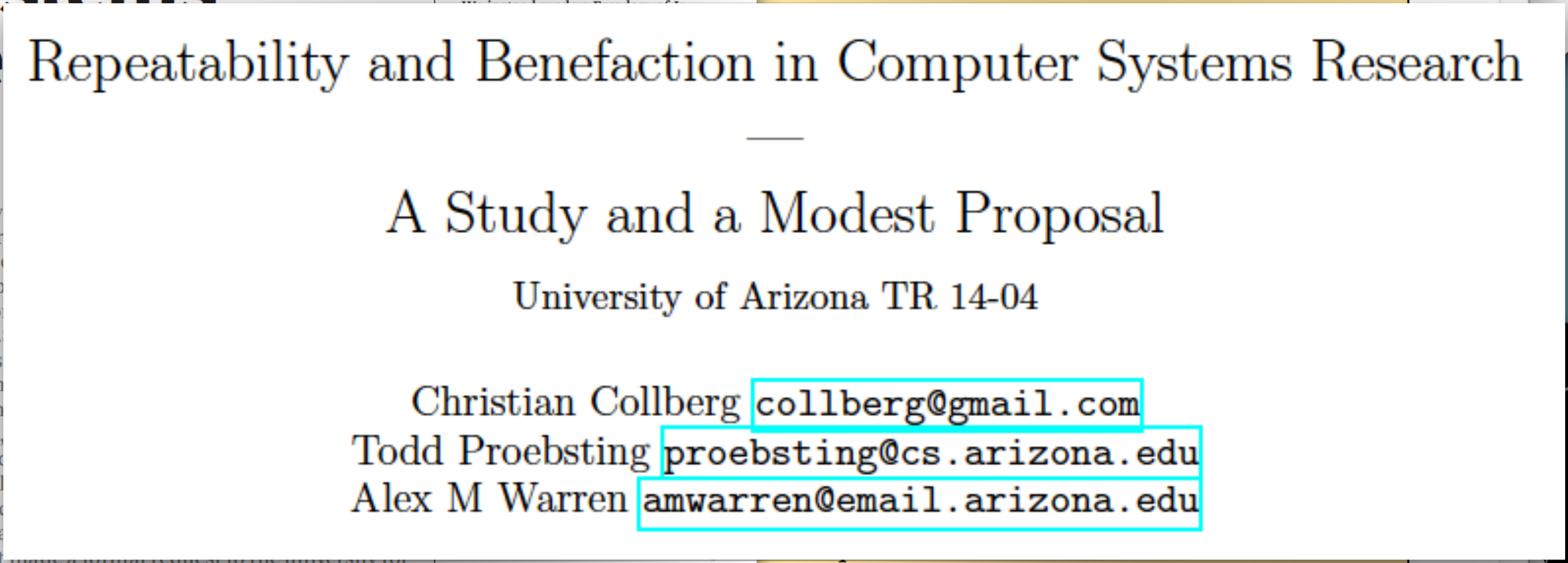
backed up, we made a second ORA request, this time for the email messages among the authors, hoping to trace the whereabouts of the source code. The legal department first responded with: "... the records will not be produced pursuant to [ORA sub-clause]." When we pointed out reasons why this clause does not apply, the university relented but demanded $2,263.66 " ... to search for, retrieve, redact and produce such records." We declined the offer.

research artifacts, publishers should require pre-publication declarations from authors specifying their commitment to sharing code and data.

ILLUSTRATION



f A group of independent researchers set out to verify our build results through a crowd-sourced effort; http://cs.brown.edu/˜sk/Memos/Examining-Reproducibility



Repeatability and Benefaction in Computer Systems Research

A Study and a Modest Proposal

University of Arizona TR 14-04

Christian Collberg collberg@gmail.com
Todd Proebsting proebsting@cs.arizona.edu
Alex M Warren amwarren@email.arizona.edu

ShriramKrishnamurthi
@ShriramKMurthi

Follow

They did ***crap*** work, would not admit to when caught out and even pretended it hadn’t happened.

https://twitter.com/ShriramKMurthi/status/863462366226370561

…these researchers have done a disservice to science by publishing a study they knew to be **horse manure**, and then piling more **bull crap** on it when caught … they are simply trying to build a reputation off a problem they don't really care to solve …

https://www.facebook.com/jvitekjr/posts/10155809013435351

To the University of Arizona Institutional Review Board:

Revoke their IRB permission!

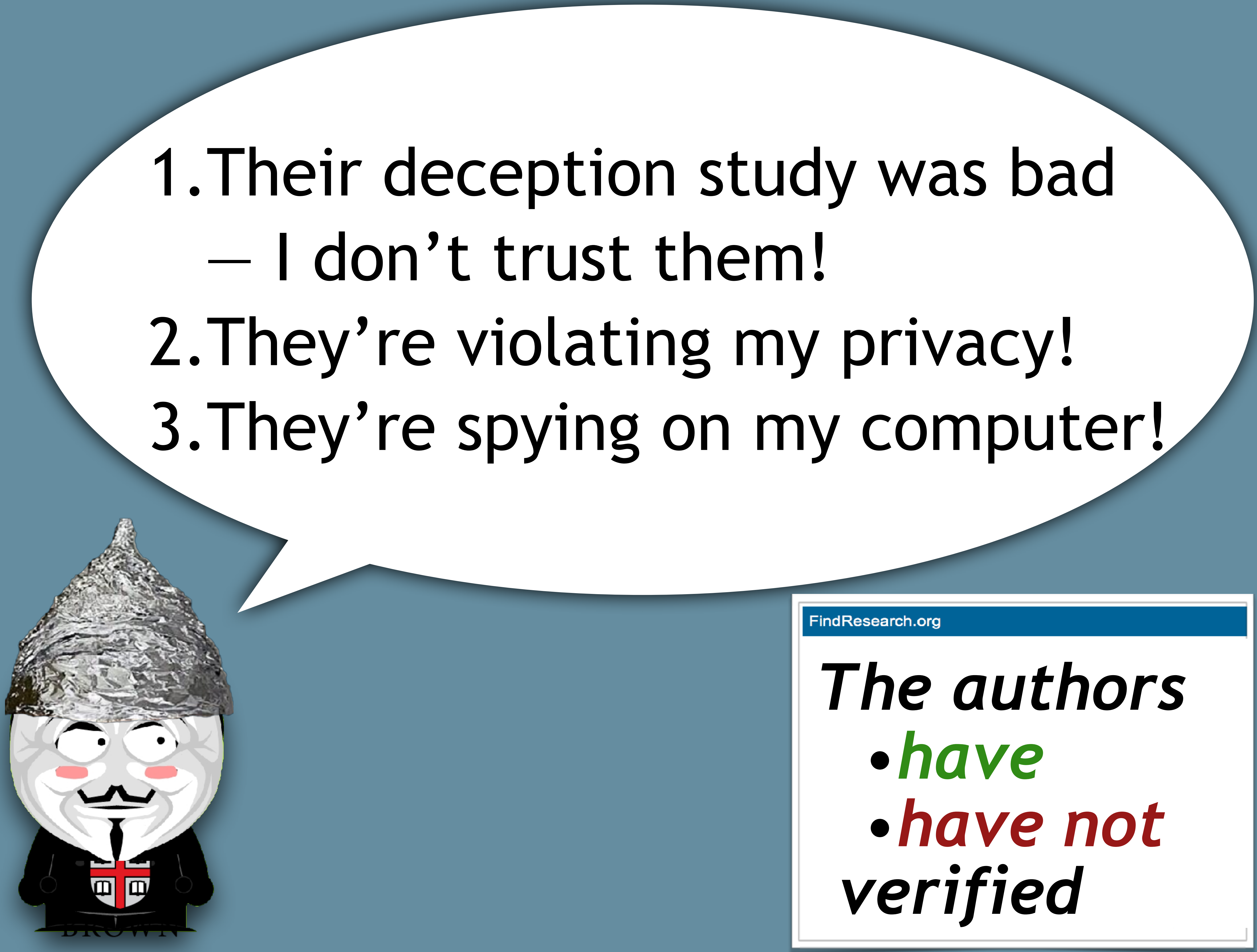
1. Their deception study was bad
   – I don’t trust them!

1. Their deception study was bad
   – I don’t trust them!
2. They’re violating my privacy!

FindResearch.org

The authors
- ***have***
- ***have not***

verified


1. Their deception study was bad
– I don't trust them!
2. They're violating my privacy!
3. They're spying on my computer!
BROWN

FindResearch.org

The authors
- *have*
- *have not*

verified

3rd Law of Artifact Sharing

(Mother's Law)

Without a culture of respectful academic interchange, where failure is seen as an accepted part of the progression of science, sharing will not become default behavior.

Risks Rewards

Risks Rewards

Credibility: They may trust your work more when they can try it.

Risks

Rewards

Credibility: They may find bugs and not trust your results.

Credibility: They may trust your work more when they can try it.

Risks

Credibility: They may find bugs and not trust your results.

Rewards

Credibility: They may trust your work more when they can try it.

Visibility: They may notice your work when they can build on it.

Risks

Credibility: They may find bugs and not trust your results.

ROI: They may ignore your code in spite of your efforts to share.

Rewards

Credibility: They may trust your work more when they can try it.

Visibility: They may notice your work when they can build on it.

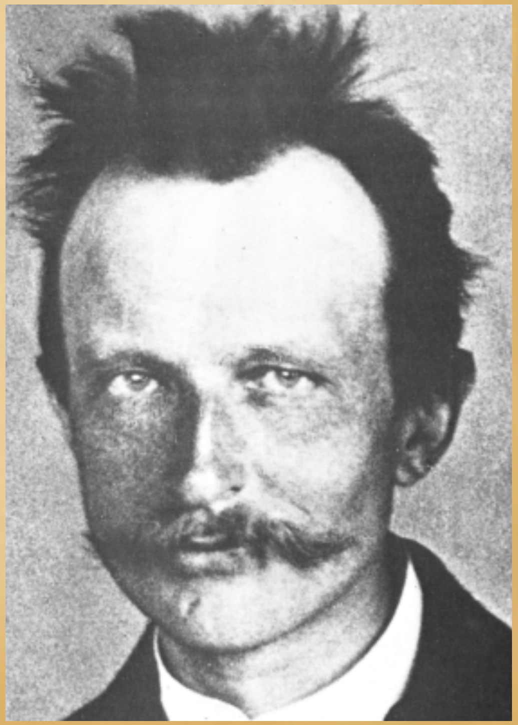
2nd Law of Artifact Sharing

The root of the scientific transparency problem is sociological, not technological: we do not share solid artifacts because there is little professional glory to be gained from doing so.

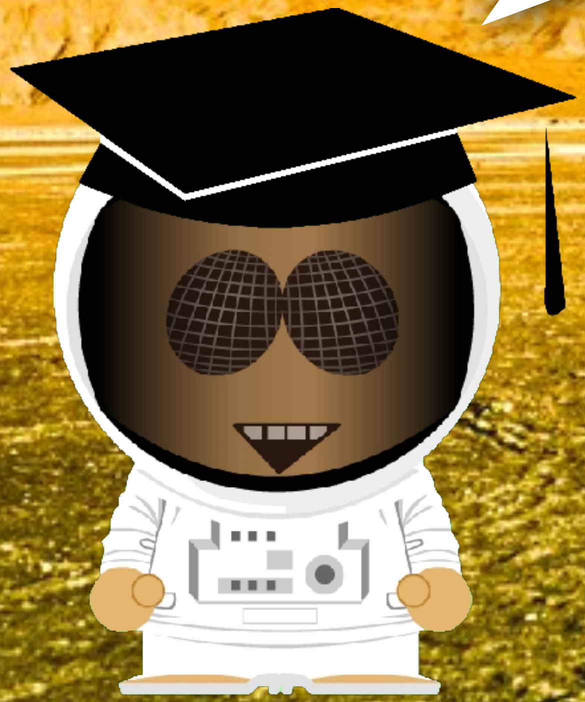
Can you believe, back in the 21st century, scientists would make up excuses why they shouldn’t have to share their research artifacts!

1st Law of Artifact Sharing
(Corollary to Max Planck's Quip)

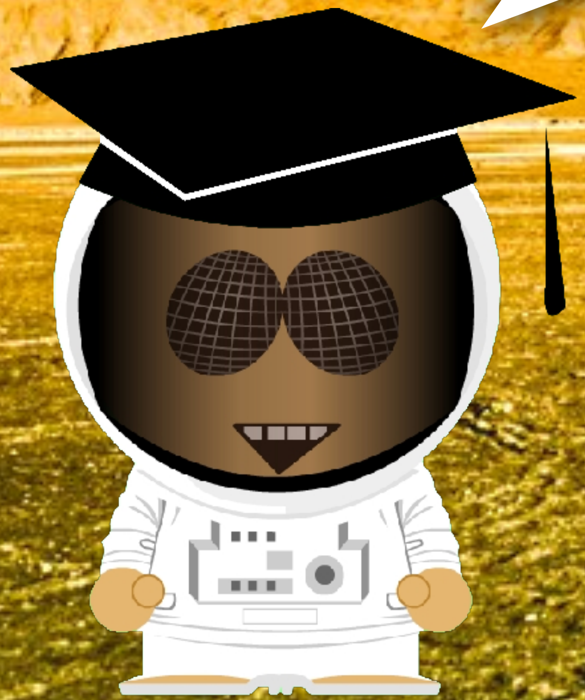
Scientific transparency advances one funeral at a time.



Educators: Propose a **Research Methods** course!

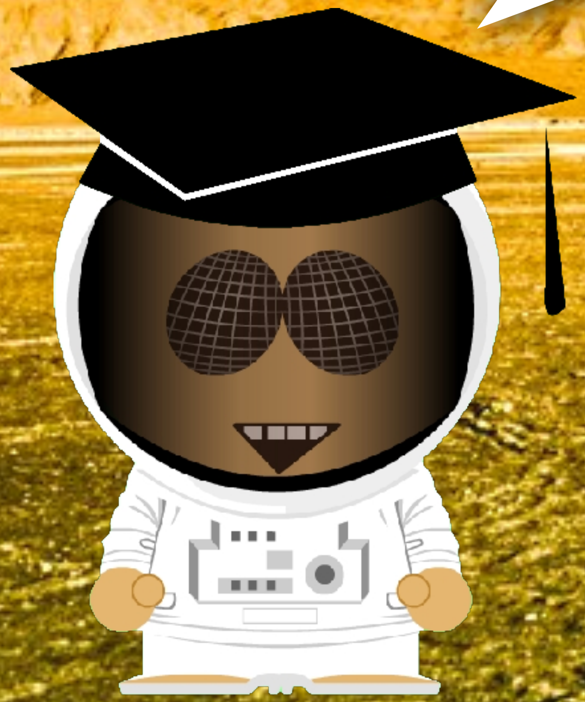


Conference chairs: Insist on **contact email** and **sharing statements**!



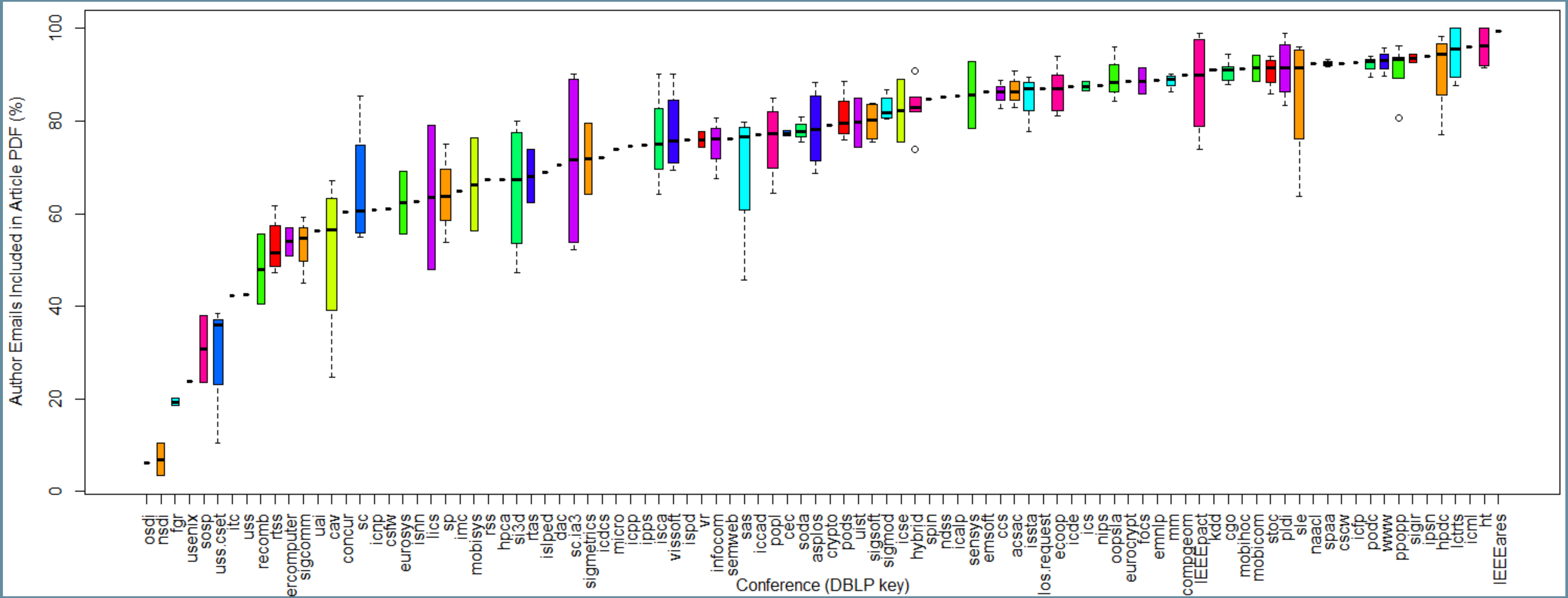
Researchers: Prepare to share and use checklists!

Funders: Require data *and* **code sharing**!

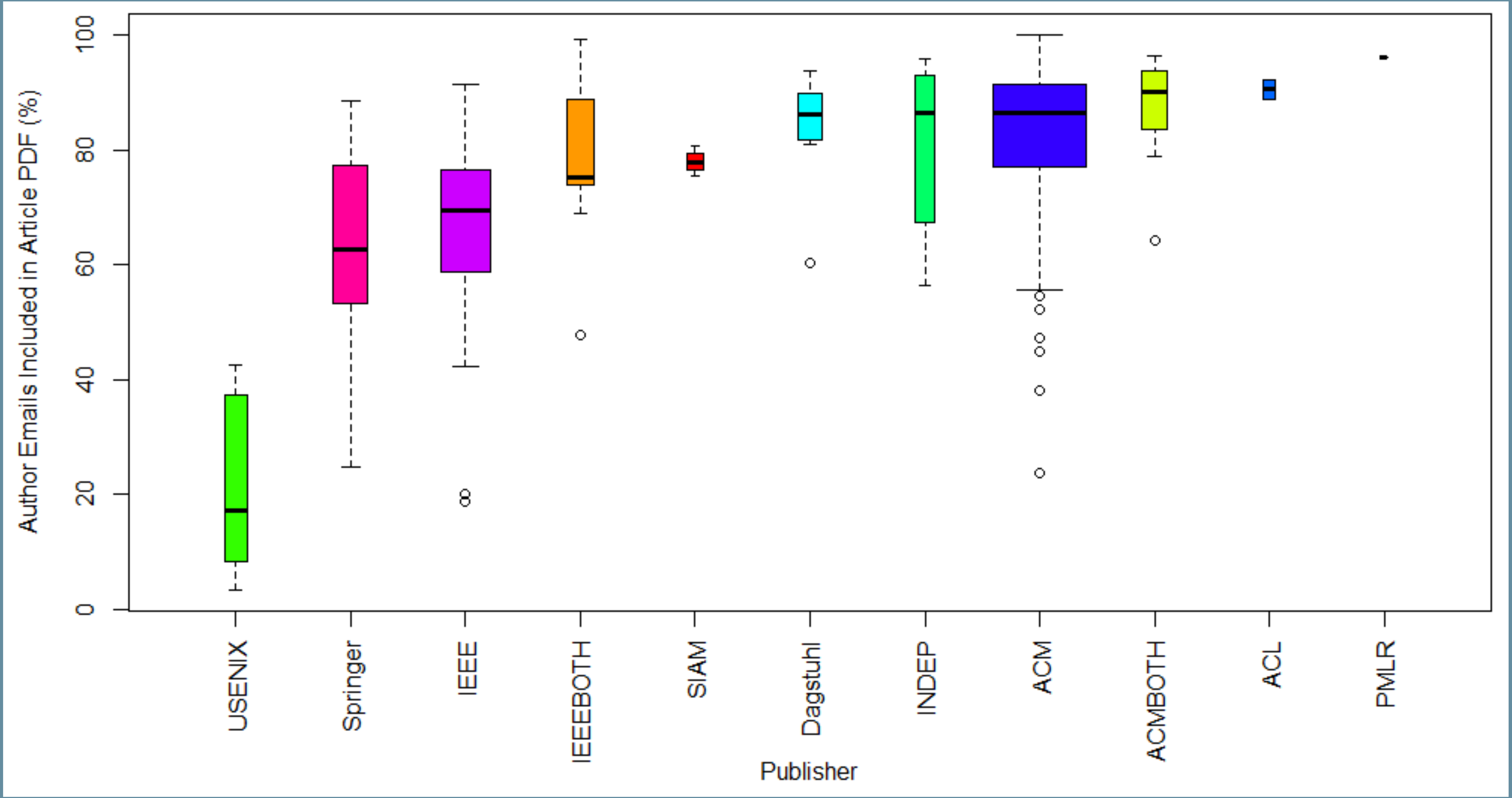


Thank you!

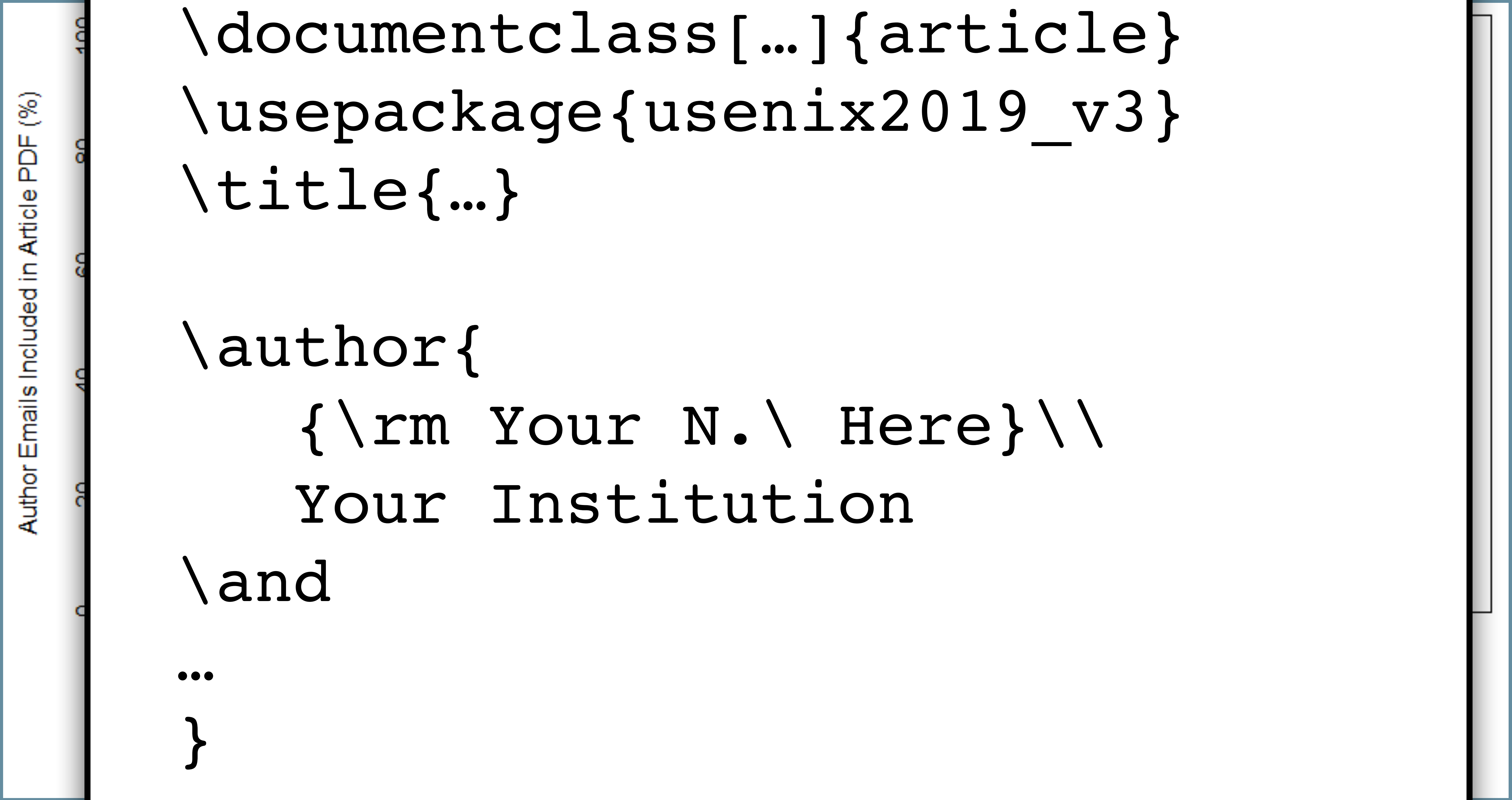
Author Emails Included in Paper



Author Emails Included in Paper



Author Emails Included in Paper



```
\documentclass[…]{article}
\usepackage{usenix2019_v3}
\title{…}

\author{
   {\rm Your N.\ Here}\\
   Your Institution
\and
…
}
```