CS 6156

Runtime Verification

Owolabi Legunsen

On the state of software quality

The New York Times

Airline Blames Bad Software in San Francisco Crash

GOOGLE SELF-DRIVING CAR CAUSED FREEWAY CRASH AFTER ENGINEER MODIFIED ITS SOFTWARE

BY JASON MURDOCK ON 10/17/18 AT 11:34 AM

Newsweek

~9% of 2017 US GDP

Report: Software failure caused $1.7 trillion in financial losses in 2017

TechRepublic.

Software testing company Tricentis found that retail and consumer technology were the areas most affected, while software failures in public service and healthcare were down from the previous year.

By Scott Matteson | January 26, 2018, 7:54 AM PST

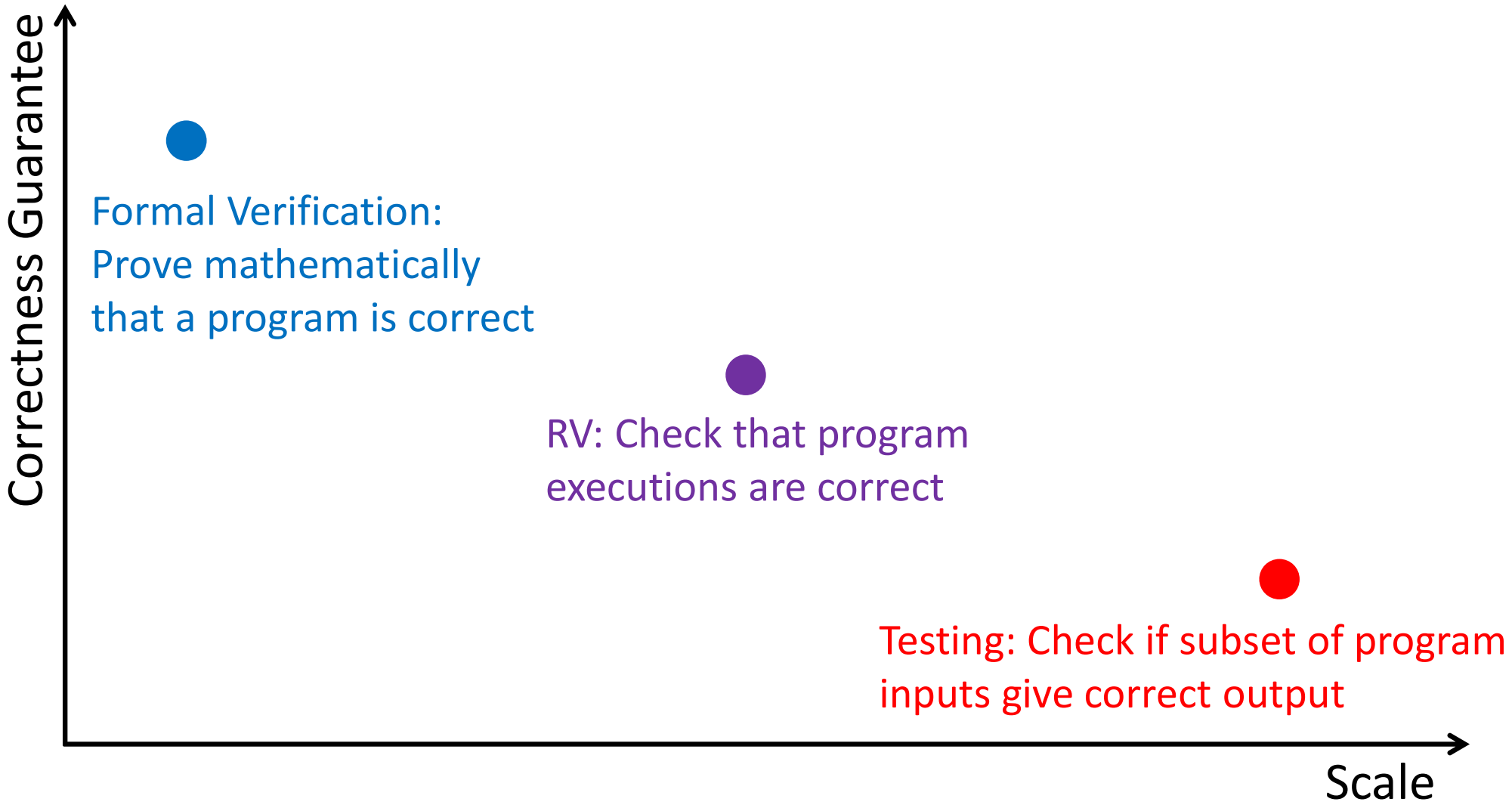
Hard Questions Raised When A Software 'Glitch' Takes Down An Airliner

Forbes

Intro to Runtime Verification (RV)

- RV is an emerging discipline for checking that ~~software~~ system executions satisfy some specifications
  - e.g., this is one of only ~3 RV courses in the world
- RV brings the mathematical rigor of formal verification to everyday ~~software~~ system development

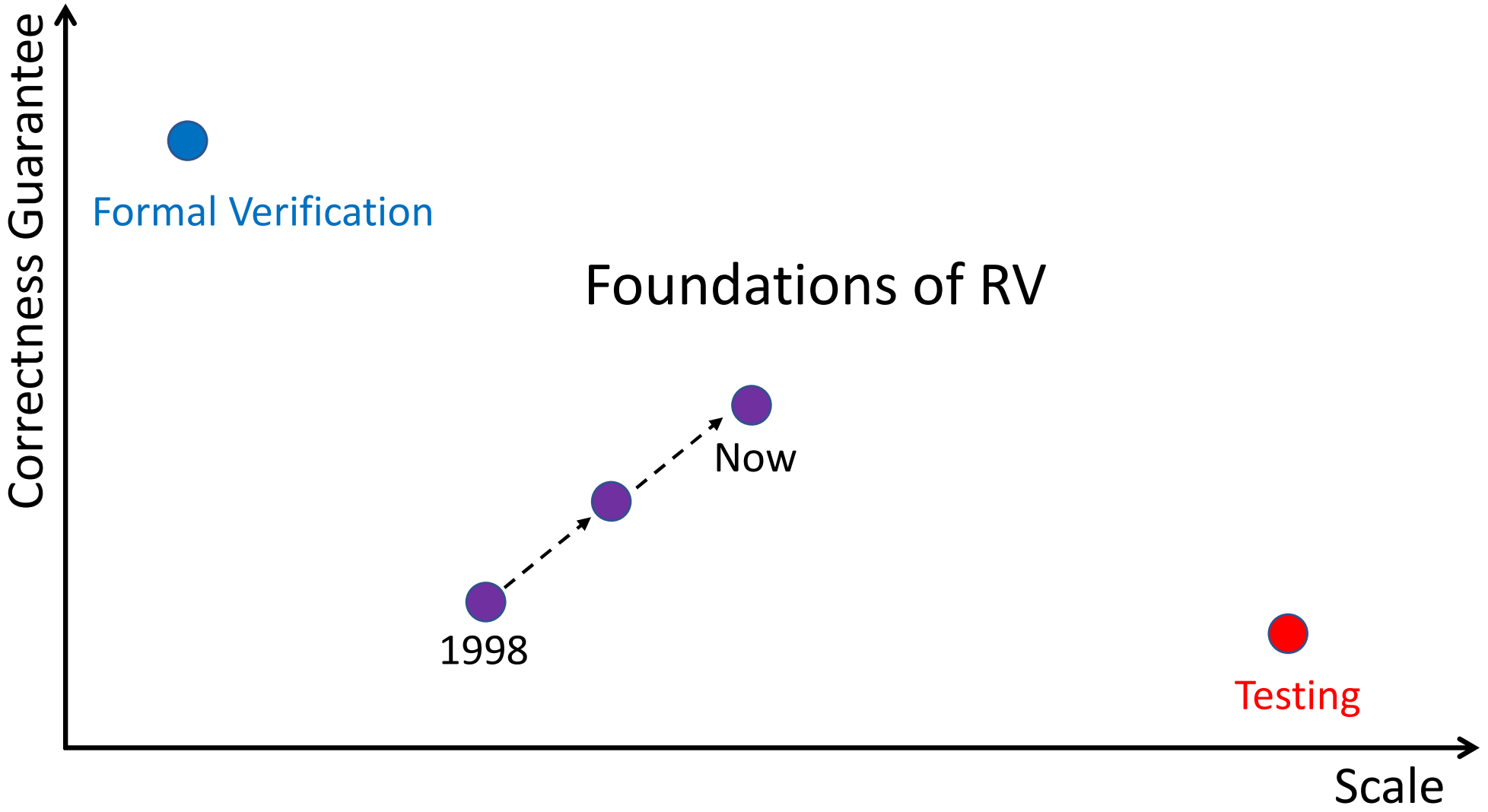
One reason why RV is appealing



About Owolabi

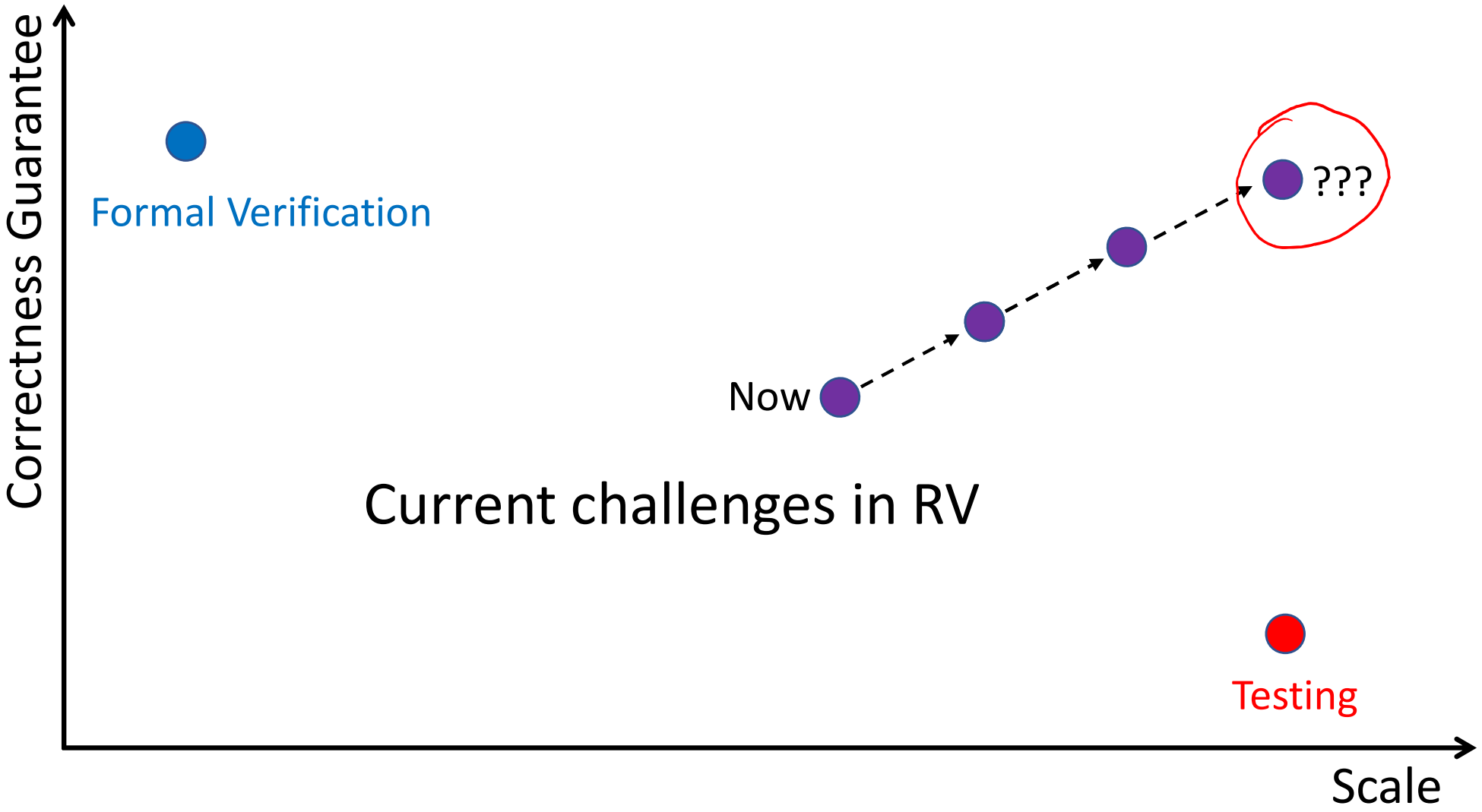
- Research interests: software testing and applied formal methods like RV
- I received my PhD from UIUC in 2019
  - thesis: incremental RV during software testing
- I found my thesis topic while trying to streamline work with my two co-advisors

What this course is about (1)



How does RV work? How to scale RV to large software?

What this course is about (2)



Can RV scale like testing and have guarantees of verification?

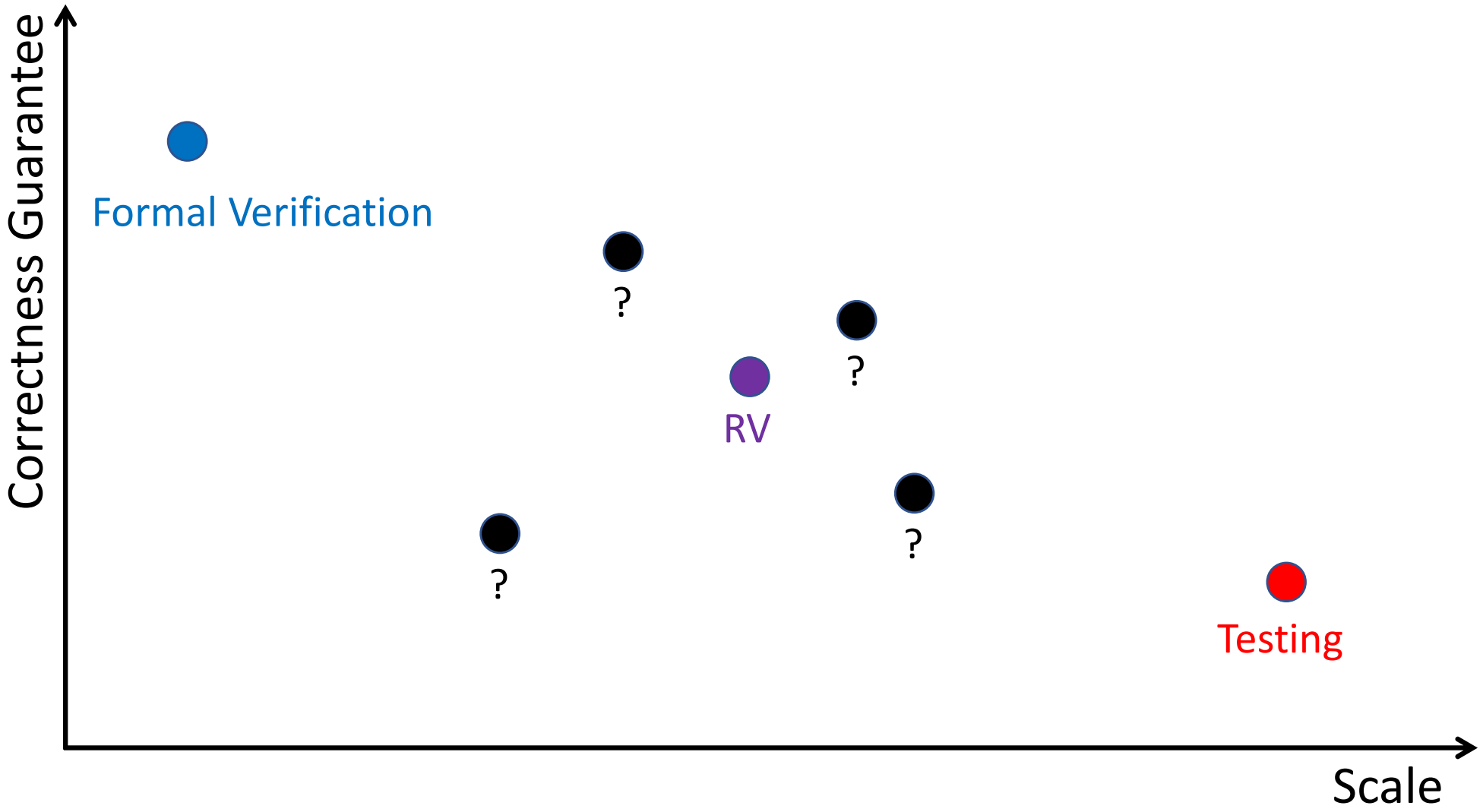
What this course is about (3)

- Hands-on exposure to RV
  - Learn how to use at least one RV tool
  - Apply RV to open-source software
  - Do research on RV or apply RV in your research (project)
  - Figure out if RV is an area of (research) interest for you

What this course is not about

- Formal verification, proof methodology, etc.
- Learning about logic (but we will use some logics)
- Basic software engineering knowledge and skills
  - Take CS5150 (Sp'22) or CS5154 (Fa'22) if that's your goal

Your turn: other QA approaches?



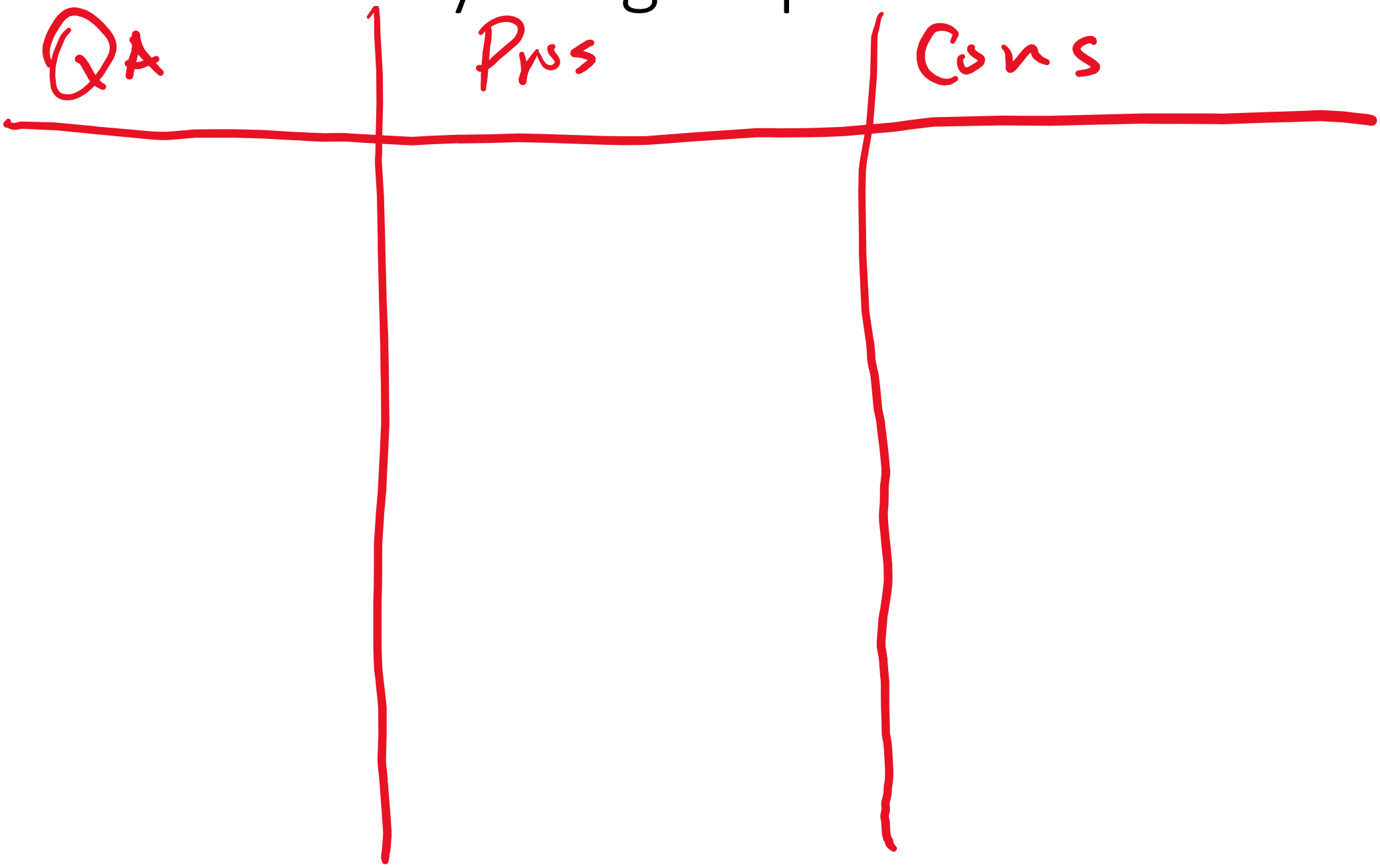
Small group discussion (10 mins)

- Introduce yourself to people in your group
- What other QA approaches have you used or heard about?
  - What are the advantages and disadvantages of each?
- Share the results of your group discussion

What did your group discuss?

QA	Pros	Cons
Symbolic Execution	• better guarantees than fuzzing • smarter exploration	state-space explosion
Code Reviews	catch style issues & other non-correctness	may (not) find bugs
Unit tests	Easy to write	not end-to-end [microservices databases] - hard to find later

What did your group discuss?



Now that we broke the ice…

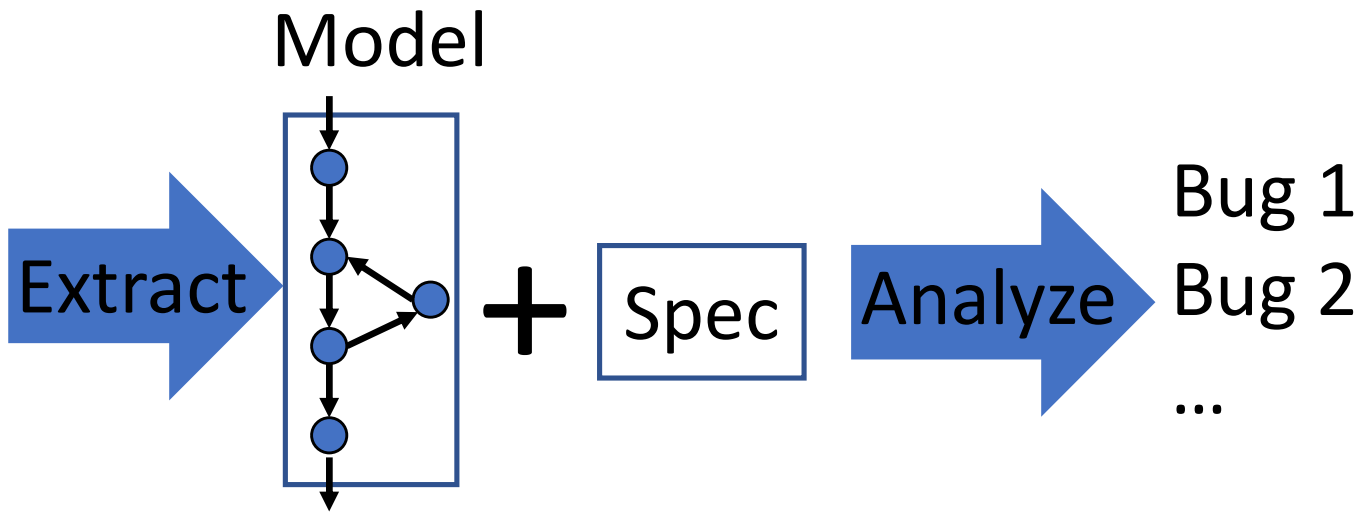
- Feel free to unmute yourself and ask questions
  - This is a small-enough discussion-based class
- Or you can post your question in the zoom chat
- At the very least, feel free to use zoom “raise hand”

Formal (static) verification

- E.g., model checking, static analysis

Code

```
int main() {
  short int a = 1024;
  int i;
  for (i = 0; i < 10; i++) {
    a *= 2;
  }
  return a;
}
```



Pros	Cons
Good code coverage	Errors in modeling
Applied early in development	False positives
Mature and well studied	Often does not scale

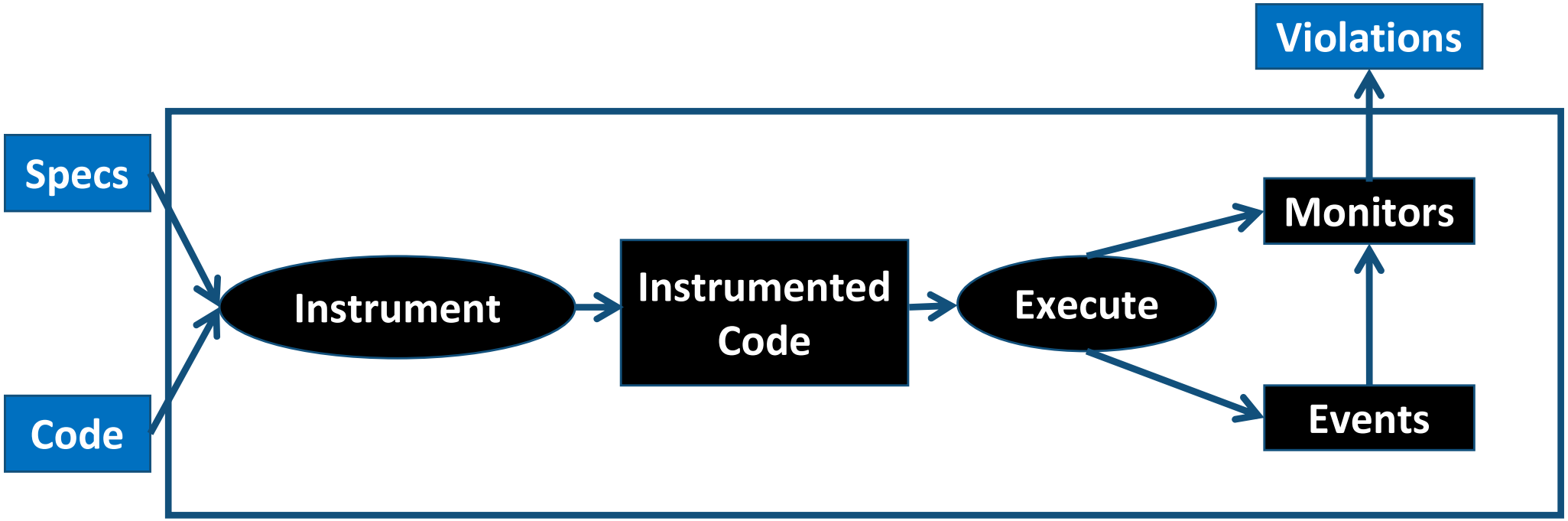
Software testing

Pros	Cons
Easier for most developers	Low code coverage (misses bugs)
Scales well in practice	Writing good oracles is hard
Leverages developer insights	High maintenance costs, e.g., obsolete tests

Runtime verification

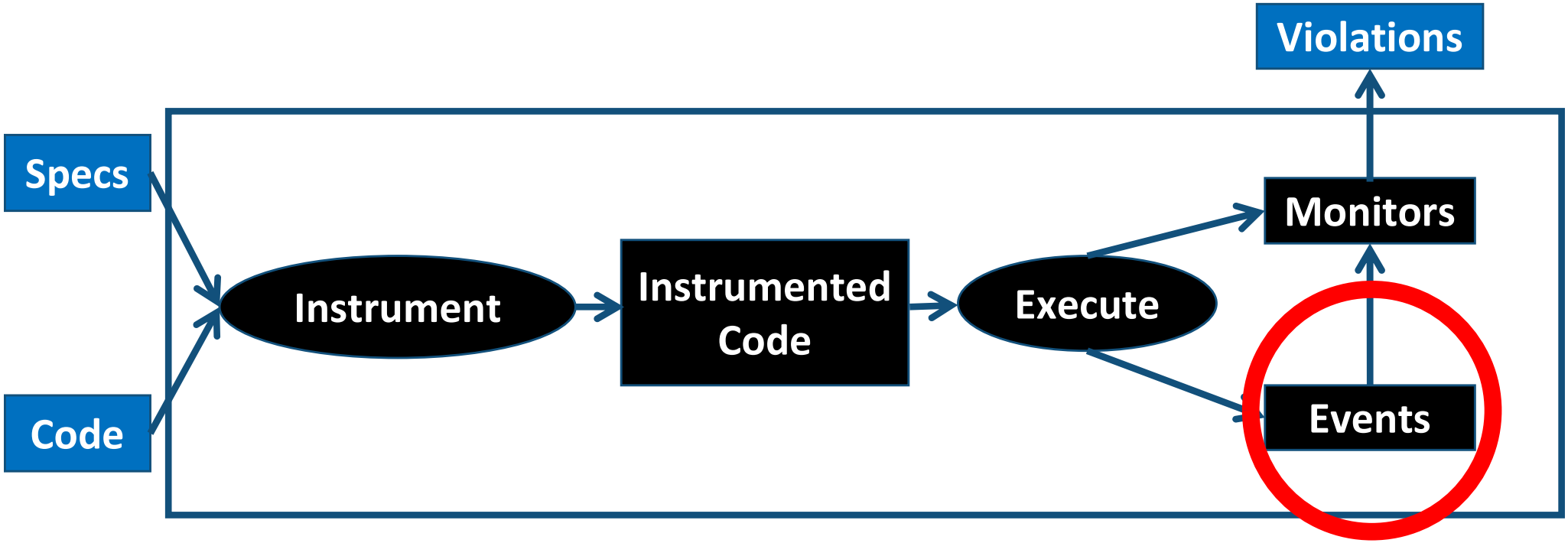
Pros	Cons
No false positives	Limited to observed executions
Scales better than formal verification	May require training in formal methods
More rigorous than software testing	More costly than software testing (higher overheads)

How runtime verification works



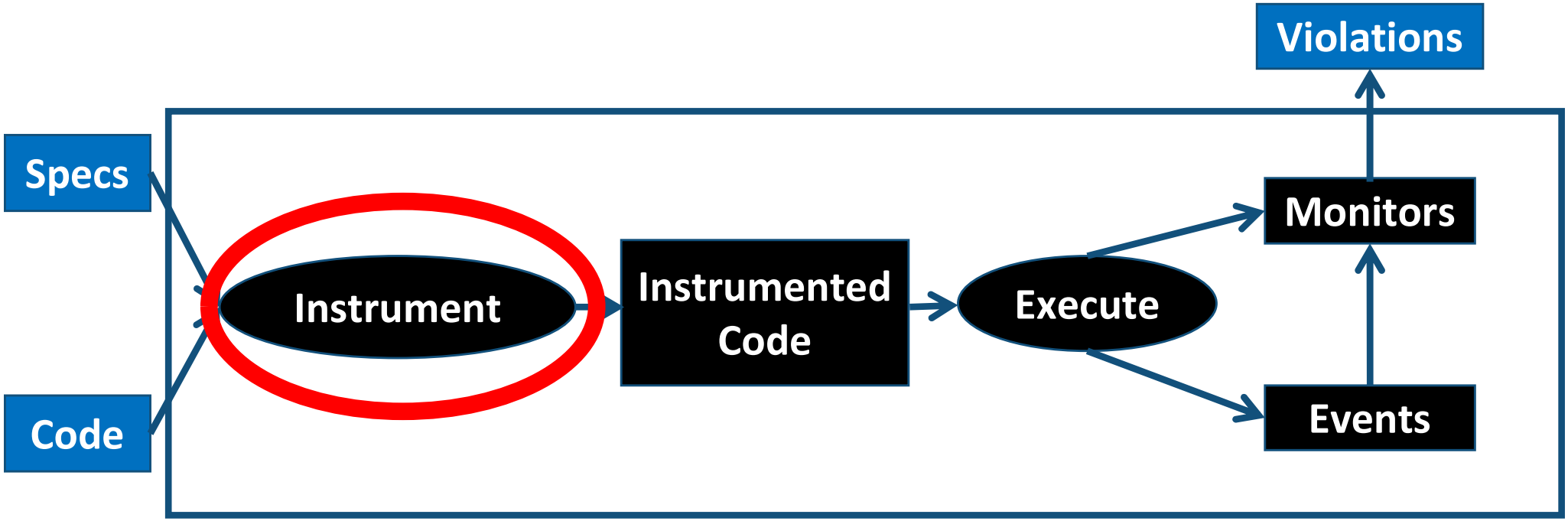
- Many (but not all) RV techniques follow this model
- CS 6156 is (mostly) organized around this model

What you'll learn (events, traces)



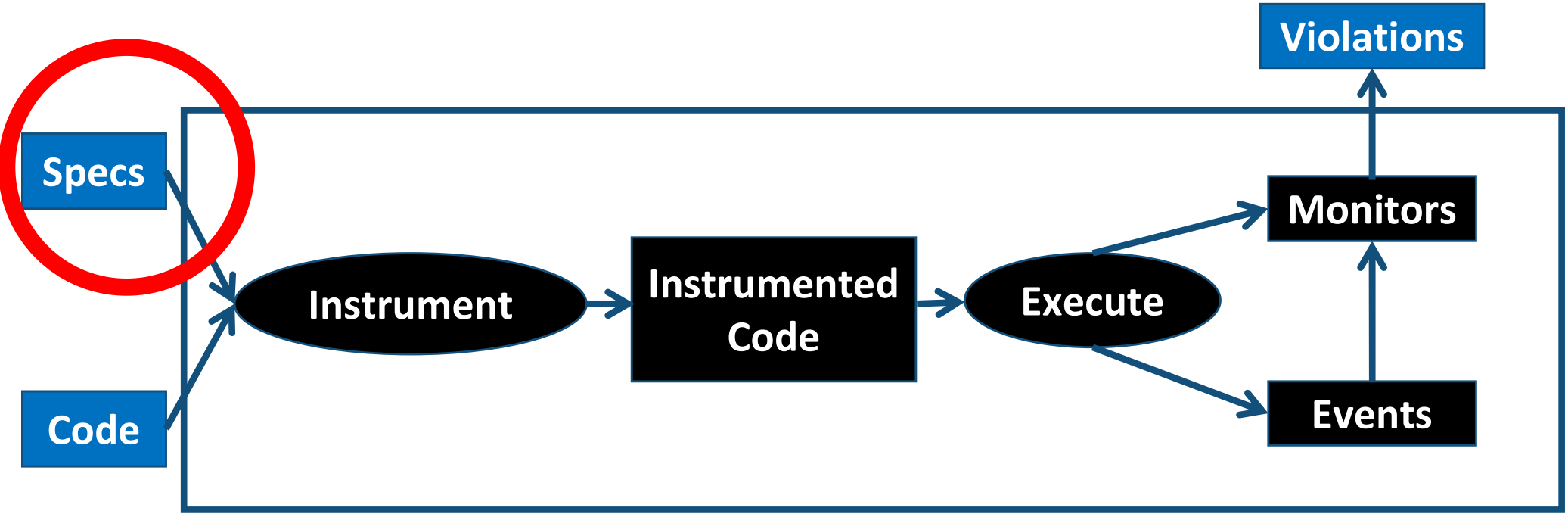
- A formal view of events, traces, and properties
- Program events (e.g., method calls, field access, etc)
- Event dispatch (e.g., which monitors to send events to?)

What you'll learn (instrumentation)



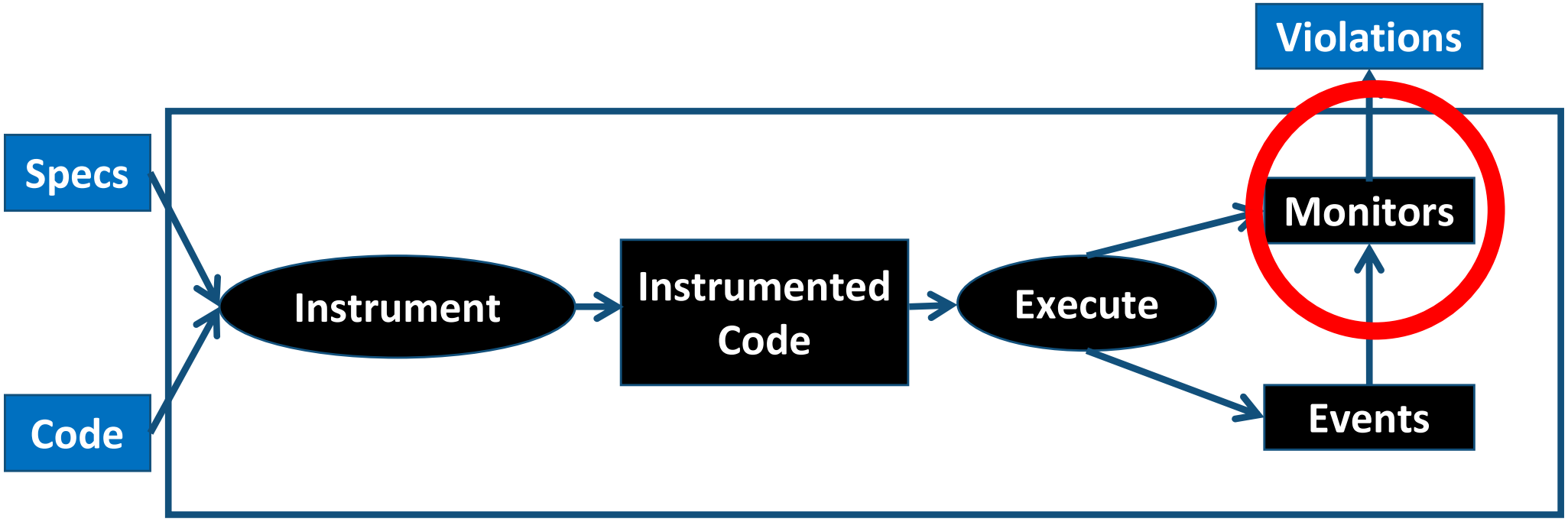
- How to instrument code to obtain runtime events?
- Compile-time vs. runtime instrumentation
- Problems and challenges of instrumentation

What you'll learn (specifications)



- What kinds of properties can RV check?
- What are languages for specifying properties in RV?
  - LTL, ERE, CFG, and other logical formalisms
- Where do properties come from? (You may write some)

What you'll learn (monitors)



- Monitor synthesis (translating specs to monitors)
- Monitoring algorithms (how monitors get and check events)
- Monitor indexing and garbage collection
  - Small-sized programs often generate tens of millions of monitors

What you'll learn (other topics)

- How to reduce RV overhead?
  - Combine with static analysis
  - Hardware-assisted RV
  - Sampling the events to check
- How to increase RV coverage?
  - Use RV during software testing
  - Incremental RV
- RV in other domains (depending on your interests)
  - Last year: hardware monitoring, networking, etc.

Is that all there is to RV?

rv22.gitlab.io

The topics of the conference include, but are not limited to:

CFP for RV 2022

- specification languages for monitoring
- monitor construction techniques
- program instrumentation
- logging, recording, and replay
- combination of static and dynamic analysis
- specification mining and machine learning over runtime traces
- monitoring techniques for concurrent and distributed systems
- runtime checking of privacy and security policies
- metrics and statistical information gathering
- program/system execution visualization
- fault localization, containment, resilience, recovery and repair
- systems with learning-enabled components
- dynamic type checking and assurance cases
- runtime verification for autonomy and runtime assurance

Application areas of runtime verification include cyber-physical systems, autonomous systems, safety/mission critical systems, enterprise and systems software, cloud systems, reactive control systems, health management and diagnosis systems, and system security and privacy.

Questions about course content?

?

Discuss: Why is RV a form of "verification"?

Compared with testing…

Is there any QA approach that can't be shown as above?

Why RV is “verification”? (vision)

1. RV can be done as a system runs in production

2. RV can allow the system to recover just before violations occur
   - Seems relatively under-explored in practice

3. So, RV can be used to ensure that a system never goes wrong with respect to a specification
   - In theory, RV can force the system to always be correct

Another reason RV is appealing

Logistics

https://freelearners.files.wordpress.com/2015/02/exploring_learning_together4-2arl9wb.jpg

CS6156 information

- Owolabi Legunsen
  - Web: https://www.cs.cornell.edu/~legunsen
  - Email: legunsen@cornell.edu
  - Office Hours: Wed 1:30-2:30pm, Fri 2:00-3:00pm

- Course web page (with in-progress schedule)
  - https://www.cs.cornell.edu/courses/cs6156/2022sp
  - Take some time to go through the web page this week
  - Check the news section frequently for announcements

You are expected to…

- Read assigned texts before each class
  - Reading for Lecture 2 is already assigned
- Complete 2-3 homework assignments
- Conduct a research project related to RV
- Lead 1 paper discussion and present your project

Your grade will be based on…

Readings and in-class participation	**10%**
Homework assignments	**5%**
Presentation and discussion lead	**15%**
Course project	**70%**

Readings

- Readings will provide deeper understanding of RV
  - **You *will* feel lost in CS 6156 if you don't read**

- Ask exactly 2-3 **non-trivial** questions on a shared PDF
  - can't ask a question that someone already asked
  - questions show that you thought about the text
  - bring other questions to class

- Due 11:59pm AOE the day before class

Presentation and discussion lead

- Each student will lead in-class discussion of a paper
  - Work with Owolabi ahead of time to prepare
  - Know the paper enough to answer classmates' questions
  - Summarize the paper in class (5-10mins)
  - Guide us in discussing questions that others asked

- Each student will also present their final project
  - (more on that in a later slide)

Homework assignments

- 2 – 3 homework throughout the semester

- Two goals
  - Assess your understanding of reading and lectures
  - Practice different aspects of RV

Projects

- Work individually or in self-assigned pairs
  - Working in pairs is strongly encouraged!
- Goal: gain deeper RV knowledge and expertise than we can cover in class + homework
- Good projects will explore research that may eventually lead to a research paper or tool

Choosing a project

- PhD Students: BYOP (bring your own project)
  - Work with Owolabi and your advisors/mentors to pick something that helps *your* research!

- BS and MEng/MS students:
  - Option 1: Owolabi can recommend some projects, but you will choose what you do
  - Option 2: Propose a research direction and explore it
  - Option 3: Pair up with a PhD student
  - Advice: avoid taking on open-ended projects alone

Tentative project timeline

Milestone	When
Discuss your project proposal with Owolabi*	Before 2/5
Project proposal is due (500 words or less)	2/8
Meet Owolabi to discuss project progress*	Before 3/5
Project progress report 1 is due (1 page or less)	3/8
Meet Owolabi to discuss project progress*	Before 4/5
Project progress report 2 is due (1 page or less)	4/8
Present final project in class*	TBD
Final project report is due (2 pages or less)	5/10

* These meetings are mandatory, but you are encouraged to meet Owolabi as often as you need!

CS 6156 is redesigned to give you more time on course project

Fall 2020	Spring 2022
Write a 500-word summary per paper	Submit 2-3 unique questions per paper
Prepare >50 slides for one paper	Lead 1 paper discussion using the submitted questions
Write ~8 pages of reports	Write ~2 pages of reports
Project was worth 50%	Project is worth 70%

Rationale: Give you more time to produce higher-quality course projects

Questions about logistics?

?

Recall: high-level view of RV

Now: concrete examples of RV tool, inputs, and outputs

- One RV tool that we will use in this class is JavaMOP
  - https://github.com/runtimeverification/javamop

Example spec: Collection_SynchronizedCollection (CSC)

https://docs.oracle.com/javase/7/docs/api/java/util/Collections.html#synchronizedCollection(java.util.Collection)

synchronizedCollection

```
public static <T> Collection<T> synchronizedCollection(Collection<T> c)
```

It is imperative that the user manually synchronize on the returned collection when iterating over it:

```
Collection c = Collections.synchronizedCollection(myCollection);
   ...
synchronized (c) {
    Iterator i = c.iterator(); // Must be in the synchronized block
    while (i.hasNext())
       foo(i.next());
}
```

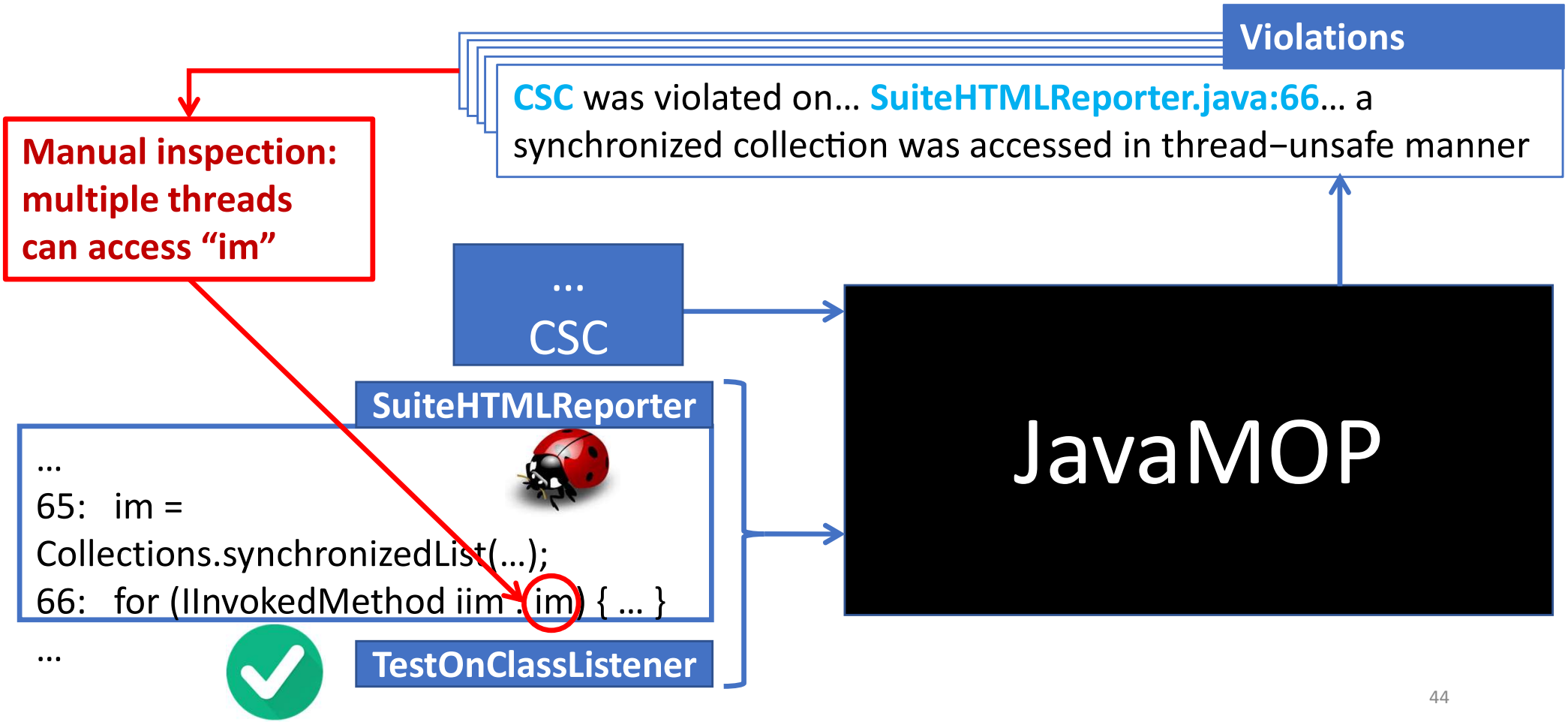
Failure to follow this advice may result in non-deterministic behavior.

Live Demo

What we saw during the demo

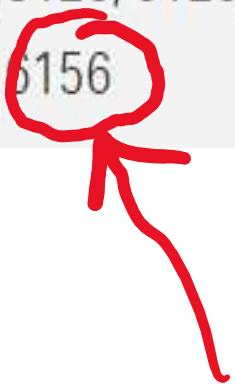
- A spec (in ERE formalism)
- JavaMOP output
- JavaMOP finds a violation in code that runs “correctly”
  - is the violation a bug, though?
- An online environment for using JavaMOP

The “RV process” (also used in demo)



CS6156: SE/PL/Systems course

	Research Styles		
	Theoretical	Systems	Applied
PL	611x, 6180	5114, 5120, 6120, 6114, 6156	6172



RV in my SE (RV + testing) research

- Monitored the tests in 229 open source software
  - some of them have over 200K lines of code
- RV found hundreds of bugs that testing missed
  - many have been confirmed
- But there are still many challenges
  - You'll discover some of them in this class

Before next class (pre-homework)

- Read the course webpage
  - https://www.cs.cornell.edu/courses/cs6156/2022sp

- If you are not a PhD student, send me an email answering these questions:
  - Your background (courses, internship, other experience)
  - What are you looking to get from CS 6156?
  - What project option are you interested in?

Next class…

- Start with the basics: events, traces, properties

- Reading is assigned (How to read SE papers)
  - Due by 11:59pm AOE Sunday 2/27/2022

What we learned so far

- A comparison of RV with other QA approaches
- A whirlwind tour of RV
- Learning outcomes, course content, and logistics
- Demo of an RV tool (JavaMOP)