

# Empirical Software Engineering

Opinion vs. Evidence in Software Development

**Dr. Sebastian Baltes**

 @s\_baltes

 [empirical-software.engineering](https://empirical-software.engineering)



# Interaction





# My Background



**Senior Software Developer**

SAP SE


Walldorf, Germany



**Adjunct Lecturer**

University of Adelaide

Adelaide, Australia



**Evidence-based Practice** through **Practice-based Evidence**



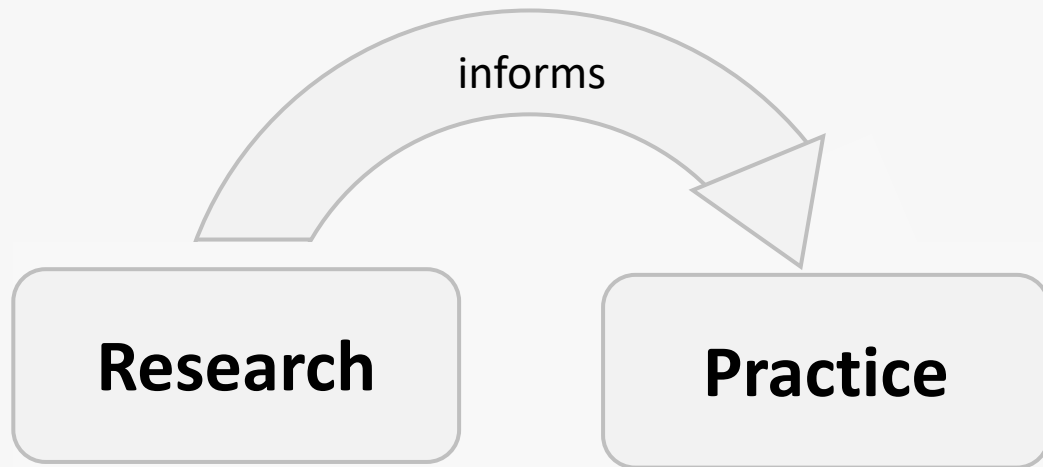
# Opinion vs. Evidence

- **Opinion:** *“Increasing test coverage reduces the number of bugs.”*
- Evidence: Wasting time testing simple code might even increase the number of bugs.  
Article 1: <https://ieeexplore.ieee.org/document/5315981>  
Article 2: <https://dl.acm.org/doi/10.1109/ESEM.2017.44>

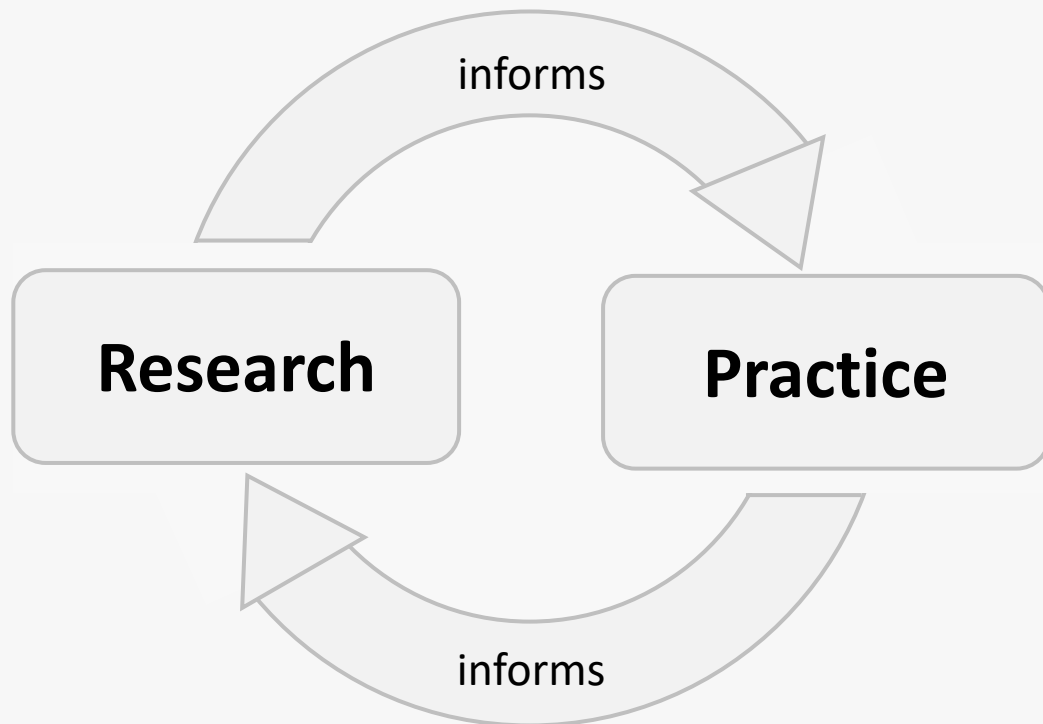
- **Opinion:** *“Test-driven development reduces number of bugs but increases development time.”*
- Evidence: Supports the above statement.  
Article: <https://link.springer.com/article/10.1007/s10664-008-9062-z>



# Evidence-based Practice through Practice-based Evidence



# Evidence-based Practice through Practice-based Evidence



## *Implications:*

- 1) Strong understanding of **state of practice** is essential
- 2) To reach this understanding, researchers need to utilize **diverse empirical research methods** and **learn from other disciplines**
- 3) To advance evidence-based practice, researchers need to **invest effort into communicating findings back to practitioners**

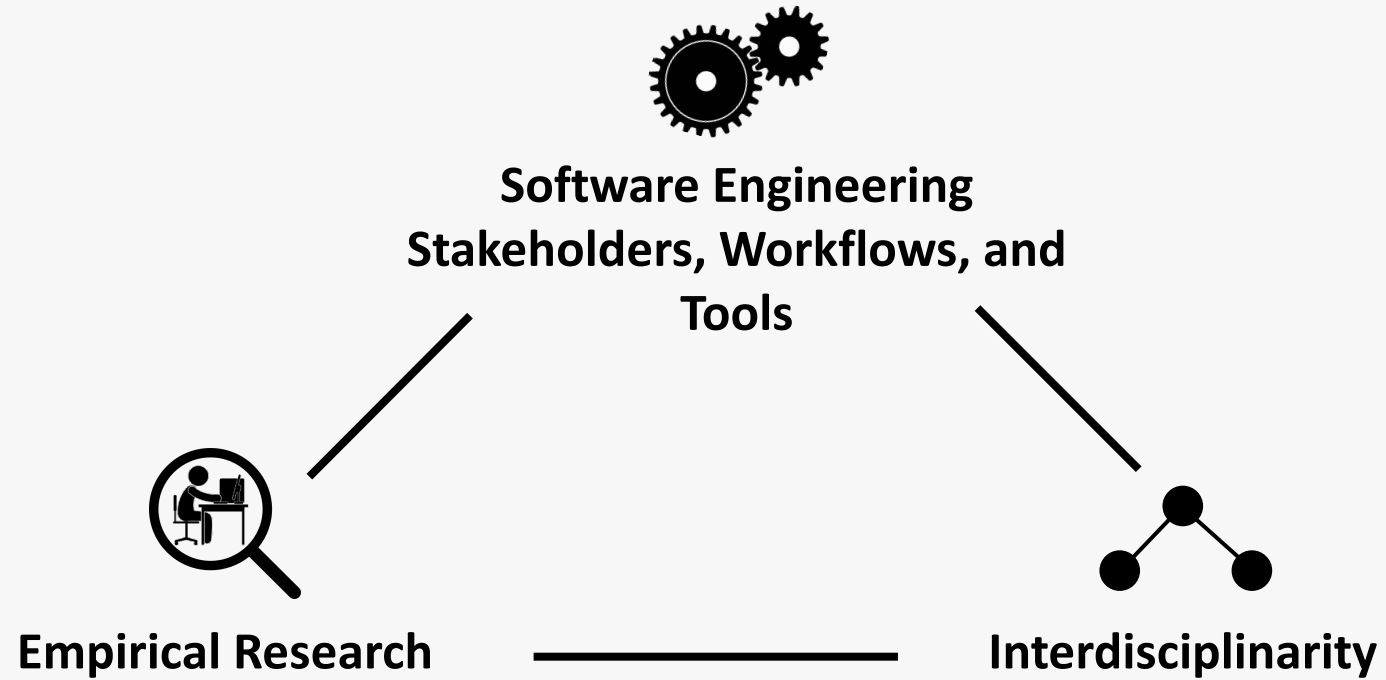


Empirical Software  
Engineering

# Empirical Software Engineering

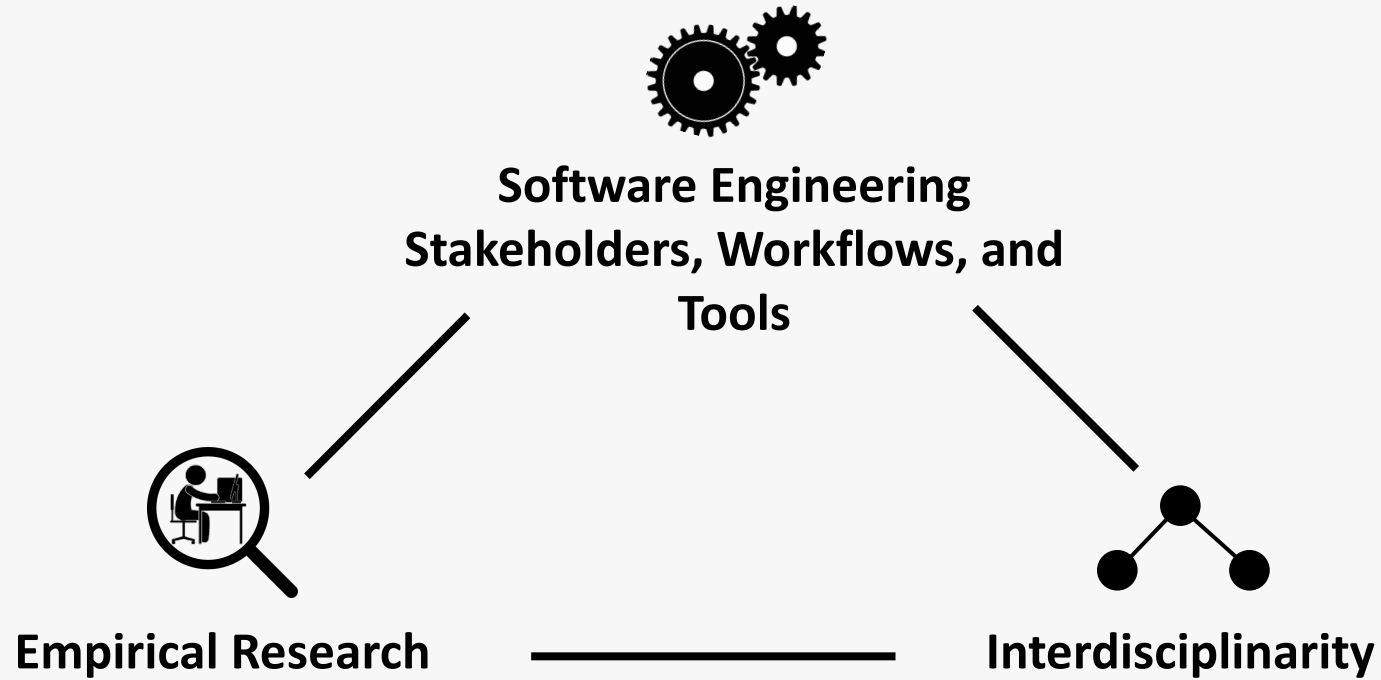
- **Software Engineering:**  
Systematically building and maintaining *software systems*
- **Software Engineering Research:**  
Systematically building and maintaining a *body of knowledge* about how to best build and maintain software systems, e.g., by exploring novel tools, process improvements, etc.
- **Empirical Software Engineering Research:**  
Software Engineering Research with a *strong empirical focus*, i.e., systematic observation/investigation of people and artifacts involved in software development

# Empirical Software Engineering

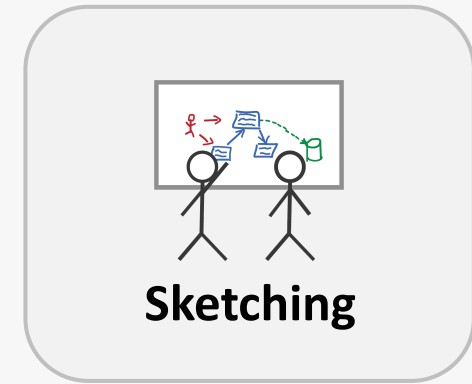
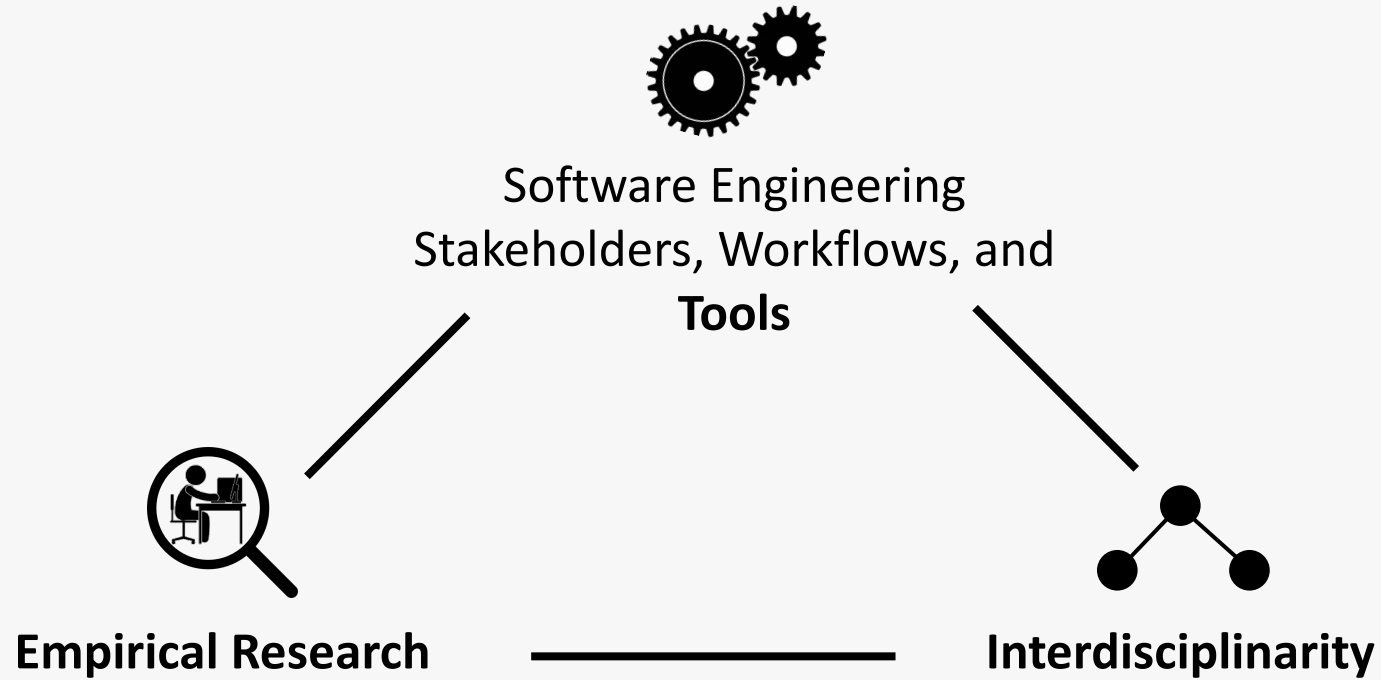




# Examples (own research)

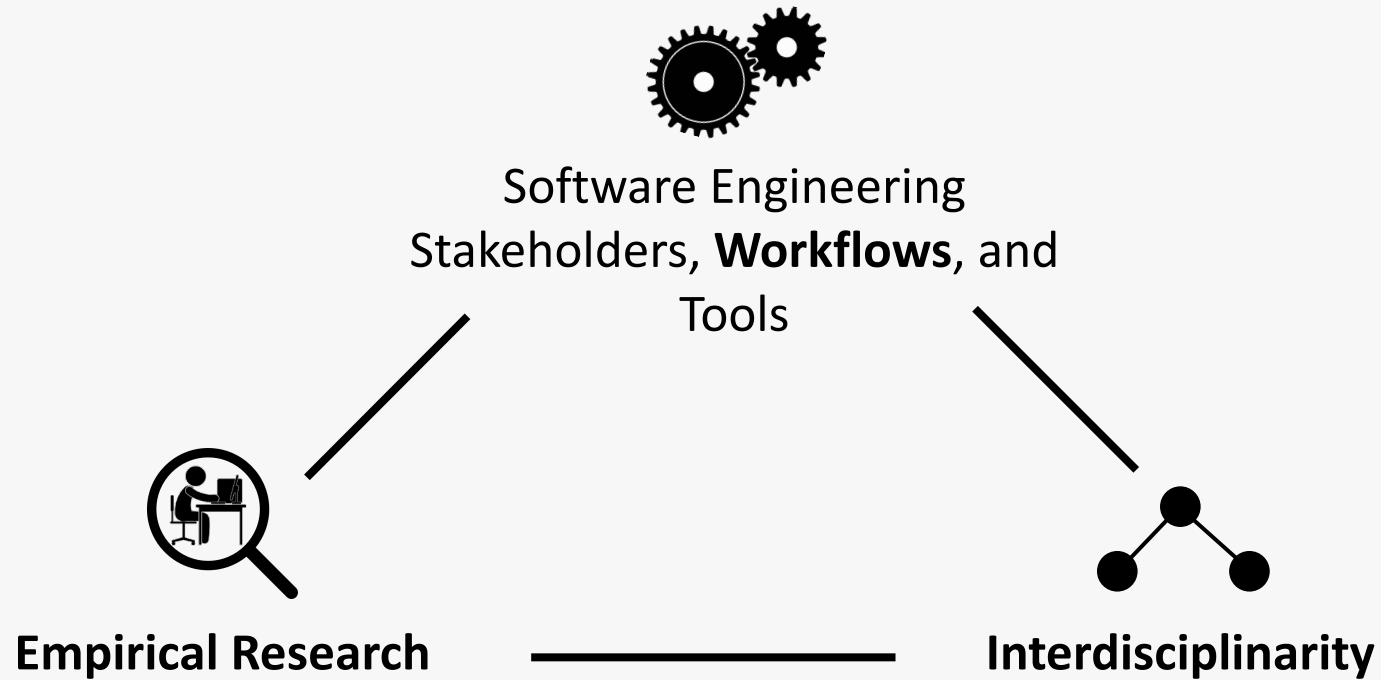


# Examples (own research)



*FSE '14,  
ESEM '15,  
VISSOFT '17*

# Examples (own research)



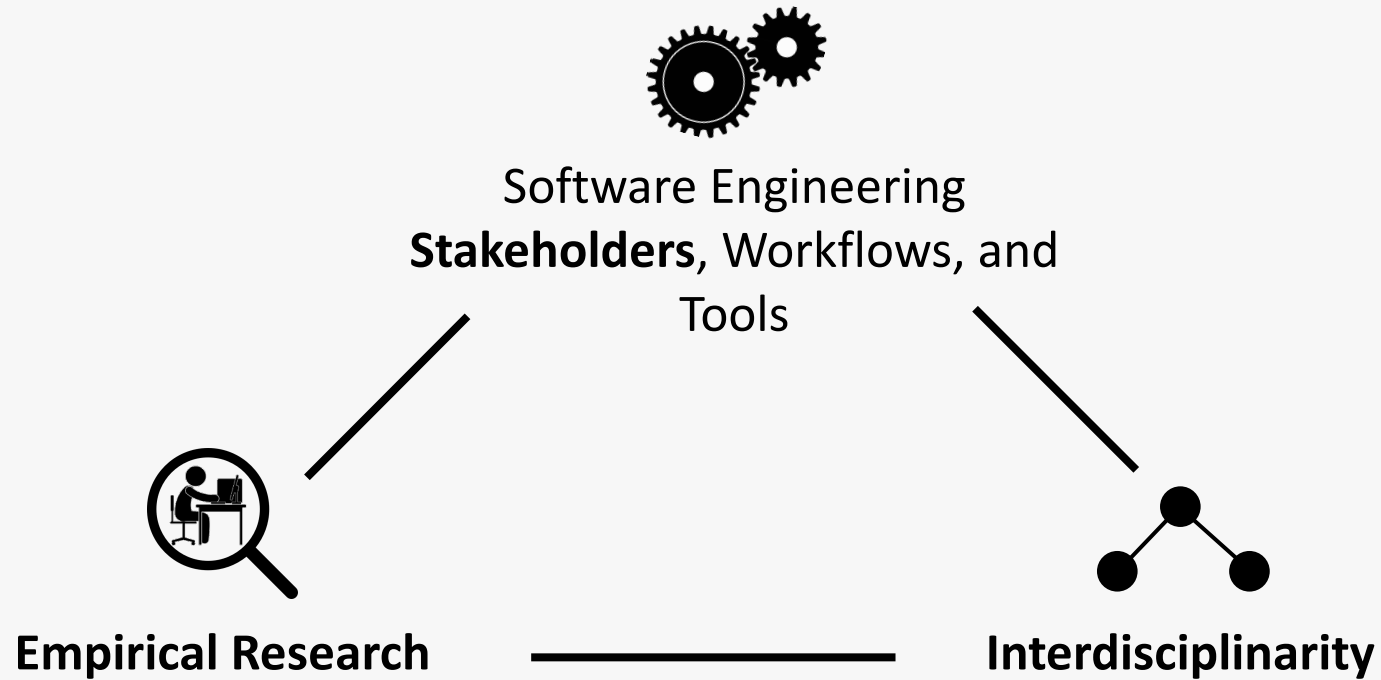
Sketching

*FSE '14,  
ESEM '15,  
VISSOFT '17*

Code Plagiarism

*EMSE '18,  
MSR '18,  
MSR '19,  
ICSE '20*

# Examples (own research)



Sketching

*FSE '14,  
ESEM '15,  
VISSOFT '17*

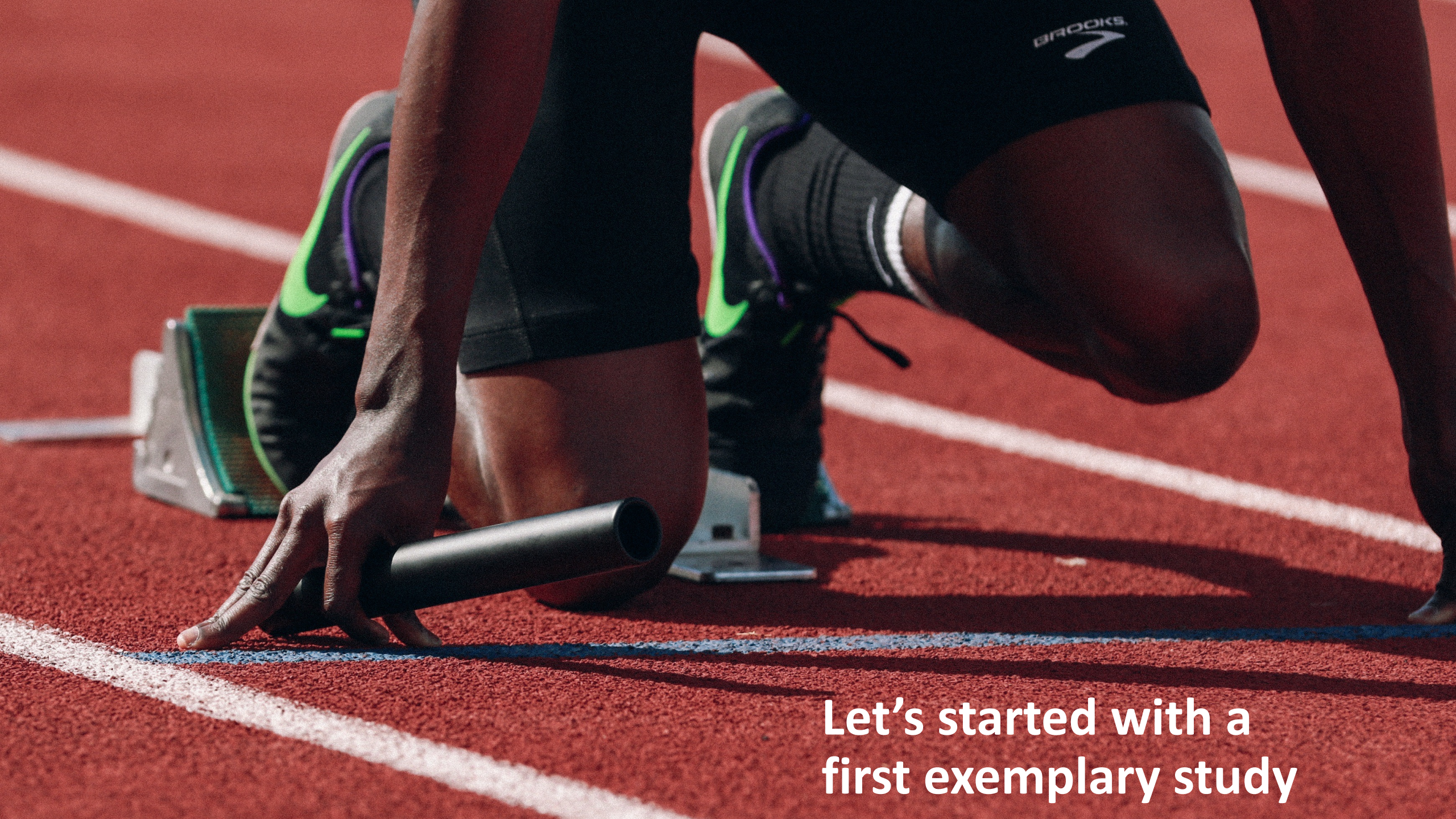
Code Plagiarism

*EMSE '18,  
MSR '18,  
MSR '19,  
ICSE '20*

Pandemic Programming

*EMSE '20*





Let's started with a  
first exemplary study



# 17. June 2021 on Hacker News

**Y** **Hacker News** [new](#) | [past](#) | [comments](#) | [ask](#) | [show](#) | [jobs](#) | [submit](#)

1. ▲ **The most copied StackOverflow snippet of all time is flawed (2019)** ([programming.guide](#))  
416 points by [vinnyglennon](#) 7 hours ago | [hide](#) | 183 comments
2. ▲ **Wayfinder – a relaxing 'art game' in the browser** ([nfb.ca](#))  
464 points by [vnglst](#) 10 hours ago | [hide](#) | 92 comments
3. ▲ **Note Taking in 2021** ([dornea.nu](#))  
97 points by [cyneox](#) 5 hours ago | [hide](#) | 49 comments
4. ▲ **Stripe Reader** ([stripe.com](#))  
70 points by [lachyg](#) 3 hours ago | [hide](#) | 58 comments
5. ▲ **Cryptanalysis of GPRS Encryption Algorithms GEA-1 suggest intentional weakness** ([iacr.org](#))  
421 points by [anonymfus](#) 13 hours ago | [hide](#) | 88 comments
6. ▲ **Bear plus snowflake equals polar bear** ([andysalerno.com](#))  
240 points by [soopurman](#) 9 hours ago | [hide](#) | 64 comments
7. ▲ **A Beginner's Guide to Miles Davis** ([samenright.com](#))  
153 points by [tinnabula](#) 9 hours ago | [hide](#) | 59 comments
8. ▲ **4-day workweek boosted workers' productivity by 40%, Microsoft Japan says** ([npr.org](#))  
348 points by [evo\\_9](#) 7 hours ago | [hide](#) | 112 comments
9. ▲ **How to Boost Self Esteem and Stop Procrastinating** ([neuralshifter.com](#))  
114 points by [CommitLock](#) 7 hours ago | [hide](#) | 54 comments
10. ▲ **Kids need freedom, too** ([persuasion.community](#))  
313 points by [jseliger](#) 12 hours ago | [hide](#) | 289 comments
11. ▲ **Why I Support the Haskell Foundation (retro on 15 years of Haskell programming)** ([cdsmithus.medium.com](#))  
28 points by [cdsmith](#) 4 hours ago | [hide](#) | 1 comment
12. ▲ **PyWhat: Identify Anything** ([github.com/bee-san](#))  
247 points by [trueduke](#) 12 hours ago | [hide](#) | 29 comments

# Code Plagiarism



*Publications:  
EMSE 2018, MSR 2018, MSR 2019, ICSE 2020 NIER*

# Code Plagiarism: Takeaways for Devs



- Software **licensing** is a **complex** topic, a general understanding of permissive vs. copyleft licenses is essential
- **Implications** of license violations for companies/individuals can be **severe**
- We can use **data mining** techniques to **detect and quantify** code plagiarism from Stack Overflow – so others can do this as well!





Here's a way using only standard Java library (note that the stream is not closed, YMMV).

2034



```
static String convertStreamToString(java.io.InputStream is) {
    java.util.Scanner s = new java.util.Scanner(is).useDelimiter("\\A");
    return s.hasNext() ? s.next() : "";
}
```

Code snippet

I learned this trick from ["Stupid Scanner tricks"](#) article. The reason it works is because [Scanner](#) iterates over tokens in ["useDelimiter\("\\A"\)](#) case we separate tokens using ["boundary" \(\A\)](#) thus giving the entire contents of the stream.

Source of snippet

Reference to JDK

**Note, if you need to be specific about the input stream's encoding, you can provide the second argument to `Scanner` constructor that indicates what charset to use (e.g. "UTF-8").**

That tip goes also to [Jacob](#), who once pointed me to the said article.

**EDITED:** Thanks to a suggestion from [Patrick](#), made the function more robust when handling an empty input stream. **One more edit:** nixed try/catch, Patrick's way is more laconic.

share in [Post edits](#)

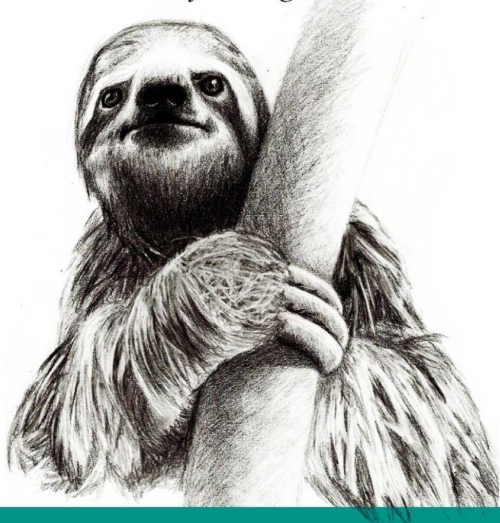
edited Sep 2 [Reasons for edits](#)

edited Mar 26 '11 at 20:40



Pavel Repin  
25.3k • 1 • 27 • 36

*Cutting corners to meet arbitrary management deadlines*



*Essential*

## Copying and Pasting from Stack Overflow

O'REILLY®

*The Practical Developer*  
*@ThePracticalDev*

## The full stackoverflow developer

*Friday, July 17th, 2015 at 1:04 pm*

In a few talks and interviews I lamented about a phenomenon in our market that's always been around, but seems to be rampant by now: the one of **the full stackoverflow developer**.

Prompted by Stephen Hay on Twitter, I shall now talk a bit about what this means.



Full Stack Overflow developers work almost entirely by copying and pasting code from Stack Overflow instead of understanding what they are doing. Instead of researching a topic, they go there first to ask a question hoping people will just give them the result.

<https://christianheilmann.com/2015/07/17/the-full-stackoverflow-developer/>

<https://twitter.com/ThePracticalDev/status/705825638851149824>



# Research Design



## Question:

How **frequently** is code from Stack Overflow posts used in public GitHub projects **without** the required **attribution**?

## Method:

**Triangulation** of an estimate for the attribution ratio using three different **data mining** approaches.

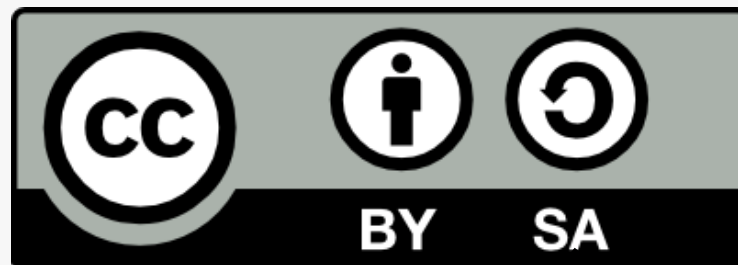
Article: <https://link.springer.com/article/10.1007/s10664-018-9650-5>



# Question for the Audience

Who knew that all content on Stack Overflow is licensed under CC BY-SA?

*"You must give **appropriate credit** [...] and indicate if changes were made."*



**Attribution**

**Share-alike**

*"If you [...] **build upon** the material, you must **distribute your contri-butions** under the same license as the original."*



# Background



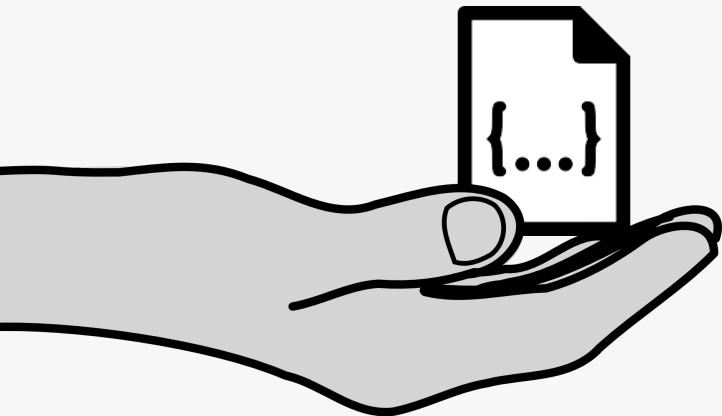
*“Well, but these snippets are rather trivial and not protected by copyright.”*

- **Not all** snippets on Stack Overflow copyrightable, but some experts argue that the **threshold is low**  
[Engelfriet 2016]
- No *“international standard for originality”*  
[Creative Commons 2017b]
- CC BY-SA is a **viral copyleft license**, affecting all modifications and derived works

# Implications of Stack Overflow's License

## Permissive Licenses

- Permit using the licensed source code in proprietary software **without publishing changes** or the **derived work**
- *Examples:* MIT, Apache, and BSD license families



## Copyleft Licenses

- Requires either modifications to the licensed content or the complete derived work to be **published under the same or a compatible license** (share-alike)
- *Examples (weak copyleft):* Mozilla/Eclipse Public Licenses
- *Examples (viral copyleft):* GNU General Public Licenses, Creative Commons Share-Alike Licenses (e.g., **CC BY-SA**)



# Enforceability of Copyleft Licenses

- Courts in the US and Europe ruled that open-source licenses are **enforceable contracts**
- Authors can **sue** when terms such as the share-alike requirement are violated:
  - **Interdict distribution** of derived work
  - **Claim monetary damages**
- USA: DMCA takedown notices for allegedly infringed copyright
  - Example: <https://github.com/github/dmca>
- Risk in mergers and acquisitions of companies
  - Example: FSF vs. Cisco lawsuit





Here's what I do:

1. First of all I check what providers are enabled. Some may be disabled on the device, some may be disabled in application manifest.
2. If any provider is available I start location listeners and timeout timer. It's 20 seconds in my example, may not be enough for GPS so you can enlarge it.
3. If I get update from location listener I use the provided value. I stop listeners and timer.
4. If I don't get any updates and timer elapses I have to use last known values.
5. I grab last known values from available providers and choose the most recent of them.

Here's how I use my class:

```
LocationResult locationResult = new LocationResult(){
    @Override
    public void getLocation(Location location){
        //Got the location!
    }
};
MyLocation myLocation = new MyLocation();
myLocation.getLocation(this, locationResult);
```

And here's MyLocation class:

```
import java.util.Timer;
import java.util.TimerTask;
import android.content.Context;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;

public class MyLocation {
    Timer timer1;
    LocationManager lm;
    LocationResult locationResult;
    boolean gps_enabled=false;
    boolean network_enabled=false;

    public boolean getLocation(Context context, LocationResult result)
    {
        //I use LocationResult callback class to pass location value from MyLocat
        locationResult=result;
        if(lm==null)
            lm = (LocationManager) context.getSystemService(Context.LOCATION_SERV

        //exceptions will be thrown if provider is not permitted.
        try(gps_enabled=lm.isProviderEnabled(LocationManager.GPS_PROVIDER);}catch
        try(network_enabled=lm.isProviderEnabled(LocationManager.NETWORK_PROVIDER

        //don't start listeners if no provider is enabled
        if(!gps_enabled && !network_enabled)
            return false;

        if(gps_enabled)
            lm.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0, locatio
        if(network_enabled)
            lm.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0, 0, loci
```

Somebody may also want to modify my logic. For example if you get update from Network provider don't stop listeners but continue waiting. GPS gives more accurate data so it's worth waiting for it. If timer elapses and you've got update from Network but not from GPS then you can use value provided from Network.

One more approach is to use LocationClient <http://developer.android.com/training/location/retrieve-current.html>. But it requires Google Play Services apk to be installed on user device.

share improve this answer edited Jun 25 '13 at 9:33 answered Jun 30 '10 at 0:07 Fedor 40k ● 9 ● 71 ● 86



```
public class MyLocation {
    Timer timer1;
    LocationManager lm;
    LocationResult locationResult;
    boolean gps_enabled=false;
    boolean network_enabled=false;

    public boolean getLocation(Context context, LocationResult result)
    {
        //I use LocationResult callback class to pass location value from MyLocation to user code.
        locationResult=result;
        if(lm==null)
            lm = (LocationManager) context.getSystemService(Context.LOCATION_SERVICE);

        //exceptions will be thrown if provider is not permitted.
        try(gps_enabled=lm.isProviderEnabled(LocationManager.GPS_PROVIDER);}catch(Exception ex){}
        try(network_enabled=lm.isProviderEnabled(LocationManager.NETWORK_PROVIDER);}catch(Exception ex){}

        //don't start listeners if no provider is enabled
        if(!gps_enabled && !network_enabled)
            return false;

        if(gps_enabled)
            lm.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0, locationListenerGps);
        if(network_enabled)
            lm.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0, 0, locationListenerNetwork);
        timer1=new Timer();
        timer1.schedule(new GetLastLocation(), 20000);
        return true;
    }

    LocationListener locationListenerGps = new LocationListener() {
        public void onLocationChanged(Location location) {
            timer1.cancel();
            locationResult.getLocation(location);
            lm.removeUpdates(this);
            lm.removeUpdates(locationListenerNetwork);
        }
        public void onProviderDisabled(String provider) {}
        public void onProviderEnabled(String provider) {}
        public void onStatusChanged(String provider, int status, Bundle extras) {}
    };

    LocationListener locationListenerNetwork = new LocationListener() {
        public void onLocationChanged(Location location) {
            timer1.cancel();
            locationResult.getLocation(location);
            lm.removeUpdates(this);
            lm.removeUpdates(locationListenerGps);
        }
        public void onProviderDisabled(String provider) {}
        public void onProviderEnabled(String provider) {}
        public void onStatusChanged(String provider, int status, Bundle extras) {}
    };

    class GetLastLocation extends TimerTask {
        @Override
        public void run() {
            lm.removeUpdates(locationListenerGps);
            lm.removeUpdates(locationListenerNetwork);

            Location net_loc=null, gps_loc=null;
            if(gps_enabled)
                gps_loc=lm.getLastKnownLocation(LocationManager.GPS_PROVIDER);
            if(network_enabled)
                net_loc=lm.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

            //if there are both values use the latest one
            if(gps_loc!=null && net_loc!=null){
                if(gps_loc.getTime()>net_loc.getTime())
                    locationResult.getLocation(gps_loc);
                else
                    locationResult.getLocation(net_loc);
                return;
            }

            if(gps_loc!=null){
                locationResult.getLocation(gps_loc);
                return;
            }
            if(net_loc!=null){
                locationResult.getLocation(net_loc);
                return;
            }
            locationResult.getLocation(null);
        }
    }

    public static abstract class LocationResult {
        public abstract void getLocation(Location location);
    }
}
```



The collage shows three GitHub repository pages. The top one is 'WuhanMonkey / MobsensAndroid' showing 'MyLocation.java'. The middle one is 'perludem / DPR-KITA' also showing 'MyLocation.java'. The bottom one is 'pacosal / ownmdm' showing 'MyLocation.java'. Red arrows point from the Stack Overflow code block to these GitHub files. A red circle highlights the 'MyLocation.java' file in each repository.





# Triangulated Attribution Ratio

*Question:* How frequently is code from Stack Overflow posts used in public GitHub projects without the required attribution?

1. **Exploratory study**
2. **Code clone detector study**
3. **Exact matches study**

}  $\bar{r}_{attr}$

We used popularity and length of the snippets as a proxy for originality and checked external availability.



# Attribution



## *Attribution ratio:*

- Method 1 (regular expressions):
- Method 2 (code clone detector):
- Method 3 (exact matches):

$$\bar{r}_{\text{attr}} = 23\%$$

$$\bar{r}_{\text{attr}} = 24\%$$

$$\bar{r}_{\text{attr}} = 8\%$$

## *Conservative estimate:*

$$\bar{r}_{\text{attr}} \leq 25\%$$

## Code Plagiarism



## Share-alike



Only **2%** of all analyzed repositories (methods 1-3) containing code from Stack Overflow **attributed** its source and used a **compatible license**.


# Reaching out to Developers



- **Contacted owners** of GitHub repositories containing copies of Stack Overflow snippets
- **75% not aware** of CC BY-SA licensing
- Many thankful responses





# Stack Overflow Code in the OpenJDK

 JDK / JDK-8170860  
Get rid of the humanReadableByteCount() method in openjdk/hotspot

---

**Details**

Type:	 Bug	Status:	<b>RESOLVED</b>
Priority:	 P2	Resolution:	Fixed
Affects Version/s:	9	Fix Version/s:	9
Component/s:	hotspot		

implement the method `humanReadableByteCount` which body was copied from the Stack Overflow site: <https://stackoverflow.com/a/3758880>

It's just a few lines of code, but it could cause legal issues. The method should be either re-implemented or removed.

Besides the potential legal issues, duplicating a code is not a good practice.

<https://bugs.openjdk.java.net/browse/JDK-8170860>

# Code Plagiarism



# Summary

**Quantification** of code plagiarism in open-source projects, outreach to developers



**Software Engineering Stakeholders, Workflows, and Tools**

**Triangulation** using three data mining approaches, online survey, (qualit. analysis)



**Empirical Research**



**Interdisciplinarity**

Research on worldwide copyright and licensing **legislation**, exemplary cases





Let's continue with  
a second example



# Code Reviews: Takeaways for Devs



- Many of the challenges around code review are **non-technical**
- **Constant (systematic) reflection** on own code review process is important
- Knowing **challenges** helps deriving solutions/mitigations  
(details later)



# Empirical SE at Microsoft: Code Reviews

*// A large-scale study of Microsoft developers revealed the challenges that code-change authors and reviewers face, best code-reviewing practices, and tradeoffs that practitioners should consider. //*

Article: <https://ieeexplore.ieee.org/abstract/document/7950877>

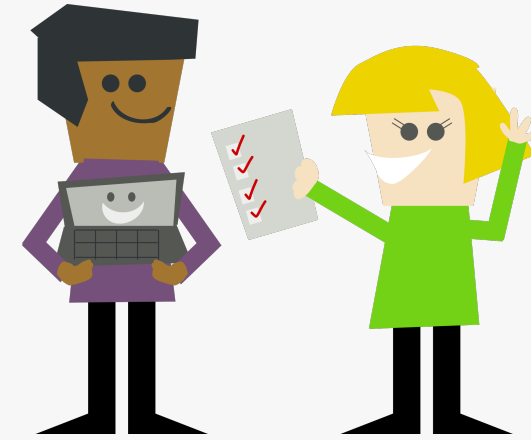


# Why study Code Reviewing?

- *“Code reviews are straight-forward to do and tool support exists, problem solved.”*
- Really? Some things to consider:
  - Level of detail (code style vs. semantic issues)
  - Code criticism turns into personal criticism
  - Large changes → LGTM
  - Code review ping pong
  - etc.
- Empirical research can help distilling **antipatterns, best practices, and requirements** for improved tool support

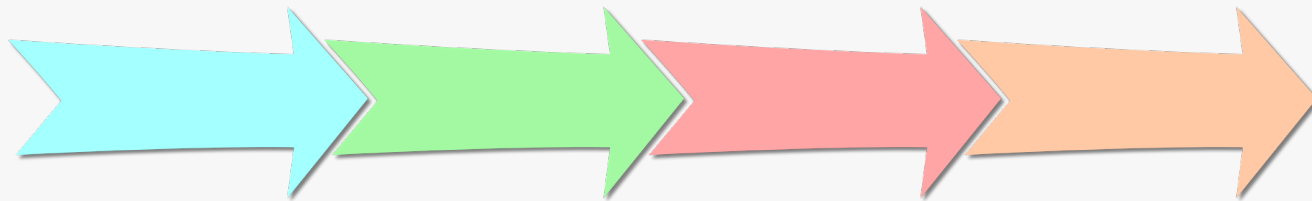
# Code Reviewing Study at Microsoft

- Focus on four teams  
(newcomers, senior developers, team leads)
- Wide range of projects  
(legacy vs. new, internal vs. external)
- Ethnographic study  
(observing developers in their workplace for one week/team)
  - Semi-structured interviews directly after code reviewing activities
  - 18 developers
- Follow-up survey with broader set of developers  
(validate initial findings)
  - 911 responses



# Code Reviewing at Microsoft

- Process (shared by all teams, internal tooling):
  - **Preparation** of code to be reviewed
  - **Selection** of reviewers (automatically or manually, varying selection requirements)
  - **Notification** of selected reviewer(s)
  - **Review** of code, sharing feedback with author(s)
  - **Iteration** (communication between authors and reviewers)
  - **Merge** code (sometimes before review)





# Code Reviewing at Microsoft

- Developers **recognize value** of code reviews
- Are **more thorough** when they know code is reviewed
- More **confidence** in reviewed code
- Not all teams had explicit **rules/policies** around code reviews

**Table 2. The respondents' code review participation.**

	During the previous week, how often did you	
	author code reviews?	act as a code reviewer?
At least once daily	17%	39%
Twice	48%	36%
Once	21%	12%
Not at all	14%	13%

# Code Reviewing at Microsoft

- **Communication** between authors and reviewers usually within tool
- **Controversial issues** discussed via other channels (face-to-face, video conference, instant messaging, etc.)  
→ no public blaming

Table 3. Motivations for code reviews\*

Reason	Overall rank
Improve code	1
Find defects	2
Transfer knowledge	3
Explore alternative solutions	4
Improve the development process	5
Avoid breaking builds	6
Increase team awareness	7
Share code ownership	8
Team assessment	9

\* The survey respondents picked and ranked their top five reasons.

# Code Reviewing Challenges: Authors

- Getting timely feedback  
(authors must constantly remind reviewers)
- Getting insightful feedback  
(focus on insignificant details rather than larger issues)
- Finding suitable/willing reviewers
- Getting a change rejected without enough feedback
- Communication in tool slows down, but other communication is often ephemeral



# Code Reviewing Challenges: Reviewers

- Reviewing large changes
- Balancing writing new code vs. reviewing others' code
- Understanding code's purpose, motivation, implementation
- Finding relevant documentation
- Lack of appreciation
- Missing training





# Code Reviewing Best Practices

Author

Reviewer

## A1. Prepare the code change for review.

A1.1. Check changes carefully.

A1.2. Aim for small, incremental changes.

A1.3. Cluster related changes.

A1.4. Describe your changes and the motivation for them.

A1.5. Test or analyze changes before submitting them for review.

A1.6. Perform a sanity check—does this change really need a review?

## A2. Select and notify reviewers.

A2.1. Decide how many reviewers you really need.

A2.2. Select reviewers with the right experience or on the basis of their need to learn the code base.

A2.3. Allow reviewers to self-select when possible.

A2.4. Check who else to notify besides the reviewers, but don't spam people.

A2.5. Notify the reviewers as early as possible and explain changes.

## A5. Respond and iterate on the change.

A5.1. Show gratitude to your reviewers.

A5.2. Be prepared to iterate on changes.

A5.3. Promote an ongoing dialogue with reviewers.

A5.4. Track the suggested changes and confirm that they're fixed.

A5.5. Generally use tools that provide decision traceability.

## A6. Commit the code change.

A6.1. Confirm that the decisions are documented.

A6.2. Reflect on the process; consider how to improve it.

## R3. Accept and conduct a code review.

R3.1. Set aside dedicated, bounded time for reviews.

R3.2. Review frequently, doing fewer changes at a time.

R3.3. Provide feedback to authors as soon as possible.

R3.4. Focus on core issues first; avoid nitpicking.

R3.5. Use or create a review checklist if necessary.

## R4. Provide feedback to the author.

R4.1. Choose communication channels carefully; talk face-to-face for contentious issues.

R4.2. Generally use tools that provide decision traceability.

R4.3. Give constructive, respectful feedback.

R4.4. Provide reasons for rejecting a change.

R4.5. Be prepared to iterate and review again.





One more thing...



# Selection of Empirical SE Courses

- University of Toronto, Canada  
<http://www.cs.toronto.edu/~sme/CSC2130/index.html>
- Carnegie Mellon University, USA  
<https://github.com/bvasiles/empirical-methods>
- University of Victoria, Canada  
<https://github.com/margaretstorey/EmseUvic2020>
- Eindhoven University of Technology, Netherlands  
[https://www.youtube.com/watch?v=34hcH7Js41I&list=PLmAXH4O57P5\\_0IfIYjLIg8l0IupZPbdIY](https://www.youtube.com/watch?v=34hcH7Js41I&list=PLmAXH4O57P5_0IfIYjLIg8l0IupZPbdIY)



**Questions?**



@s\_baltes



empirical-software.engineering

**Dr. Sebastian Baltes**