# Investigating the Validity of Ground Truth in Code Reviewer Recommendation Studies

**Emre Doğan**, Eray Tüzün, K. Ayberk Tecimer, H. Altay Güvenir

MSc Student
Computer Engineering Department
Bilkent University

ESEM 2019
Porto de Galinhas, Brazil
19th August 2019

# Outline


1. Introduction
   - Code Review and Code Reviewer
   - Code Reviewer Recommendation
   - Reviewer Selection
2. Ground Truth
   - Definitions
   - Problematical Reviewer Selection Scenario
   - Ideal Reviewer
3. Reasons of Non-Ideality
   - Availability Reasons
   - Cognitive Bias
4. Possible Reviewer Assignment Scenarios
5. Quantitative Evidence
6. Solution Alternatives
   - Expensive Setup in Real Life
   - Forward-Looking Mining
7. Conclusion & Future Work
8. References

# What is Code Review, Who is a Code Reviewer?



**Code Review:** A systematic examination of source code in order to highlight bugs and enhance the code quality.

**Code Reviewer:** The developer performing a code review.

# What is Code Review, Who is a Code Reviewer?



**Code Review:** A systematic examination of source code in order to highlight bugs and enhance the code quality.

**Code Reviewer:** The developer performing a code review.

# What is Code Review, Who is a Code Reviewer?



**Code Review:** A systematic examination of source code in order to highlight bugs and enhance the code quality.

**Code Reviewer:** The developer performing a code review.

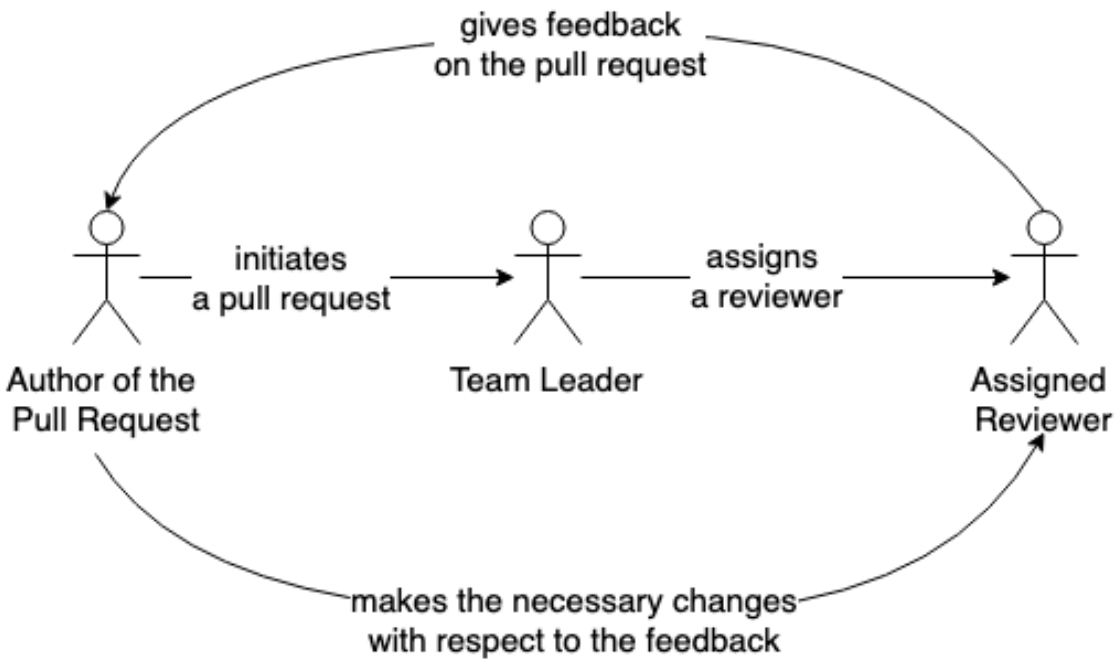



Figure: A typical code review scenario

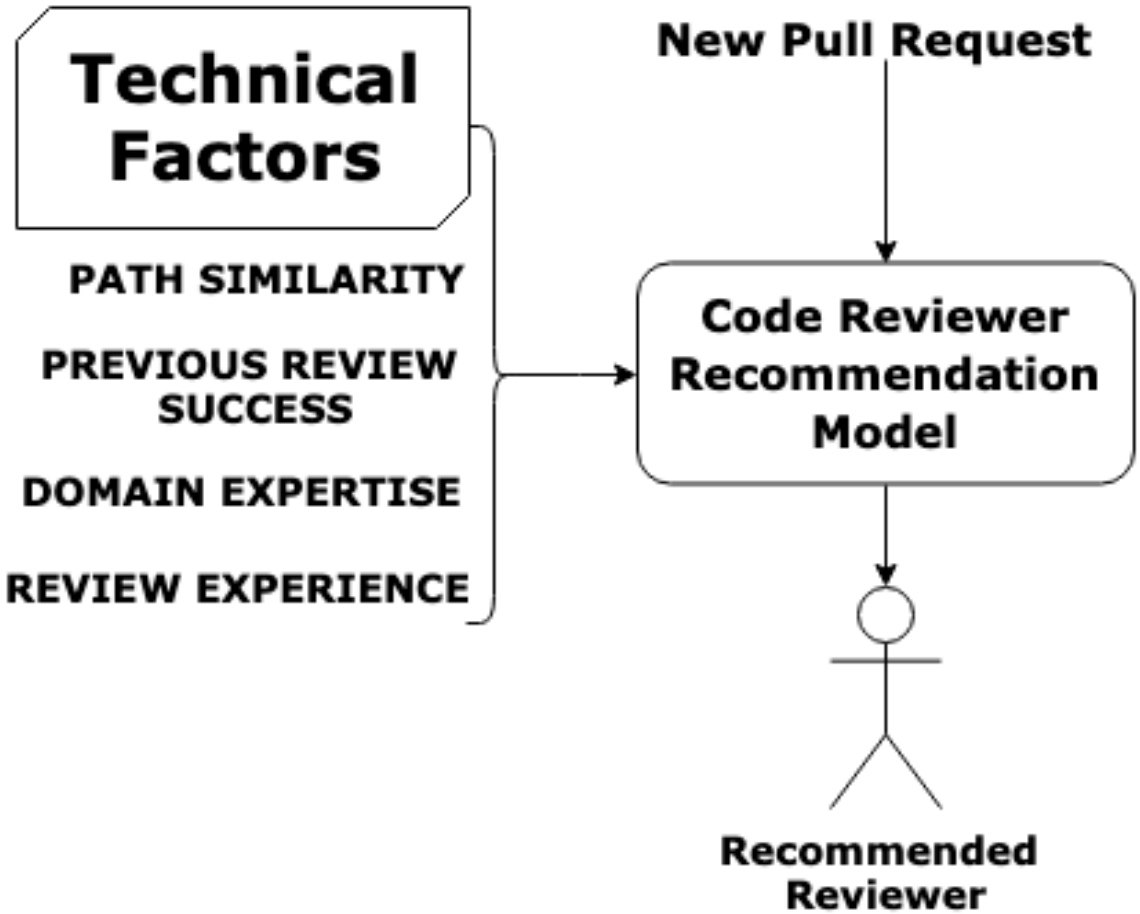
# How to find an efficient code reviewer?



- Code reviewer recommendation models/tools help us to choose efficient reviewers.
- These tools help software teams:
  - to find reviewers who can find more(critical) bugs in the source code.
  - to speed up the code review process.

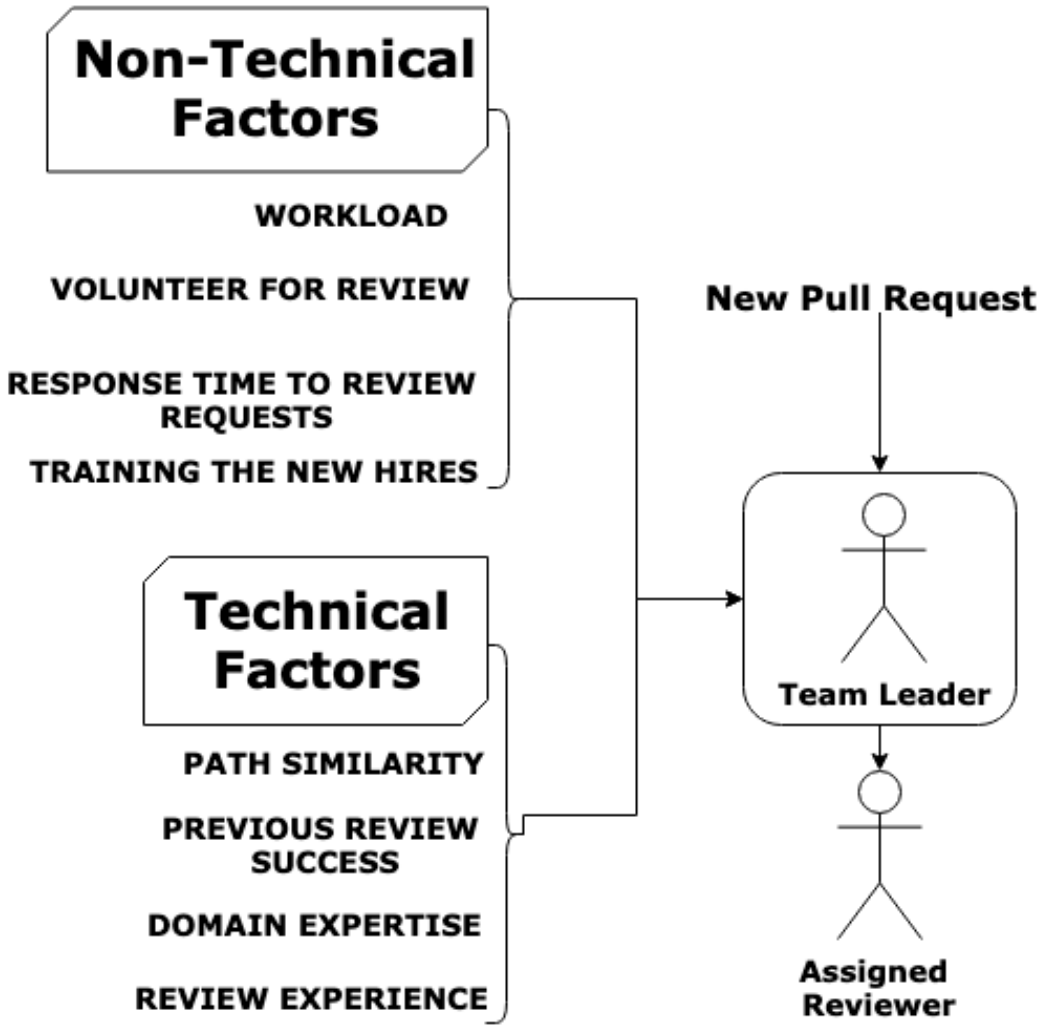
# Reviewer Selection in Recommendation Models





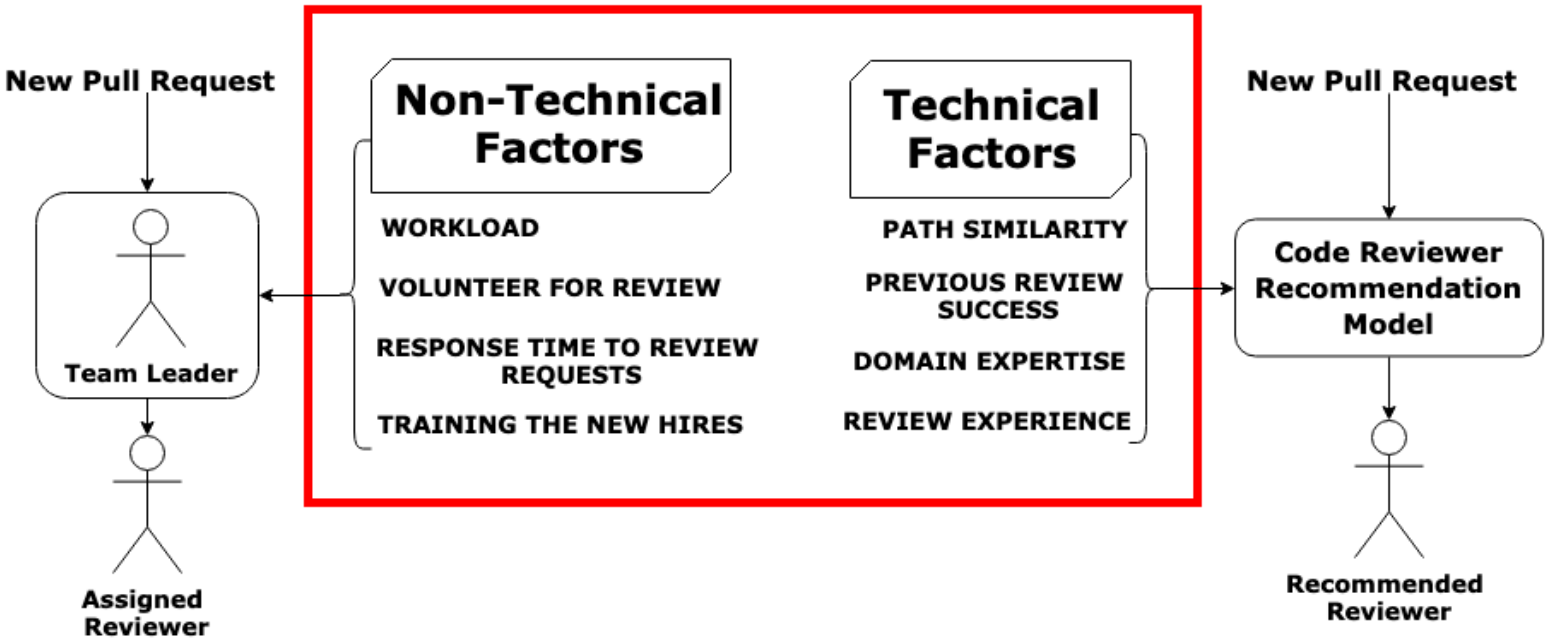
# Reviewer Selection in Real Life





# Comparison of Real Life and Algorithms






**Notice:**

- There exists a discrepancy between real life and algorithm based reviewer selection process.
- This discrepancy creates a **ground truth problem** in code reviewer recommendation studies and datasets.

# Ground Truth



## Ground Truth:

- Factual data that has been observed or measured.
- If data stands on some assumptions, is subject to opinion, then it cannot be **ground truth data**.

## Ground Truth in Software Engineering:

- The more human aspects involved, the more tendency to the ground truth problems.
- Many fields of empirical software engineering research suffer from the ground truth problem. (i.e. code reviewer recommendation, bug report assignee recommendation, etc.)

# Ground Truth



## Ground Truth:

- Factual data that has been observed or measured.
- If data stands on some assumptions, is subject to opinion, then it cannot be **ground truth data**.

## Ground Truth in Software Engineering:

- The more human aspects involved, the more tendency to the ground truth problems.
- Many fields of empirical software engineering research suffer from the ground truth problem. (i.e. code reviewer recommendation, bug report assignee recommendation, etc.)

# Ground Truth

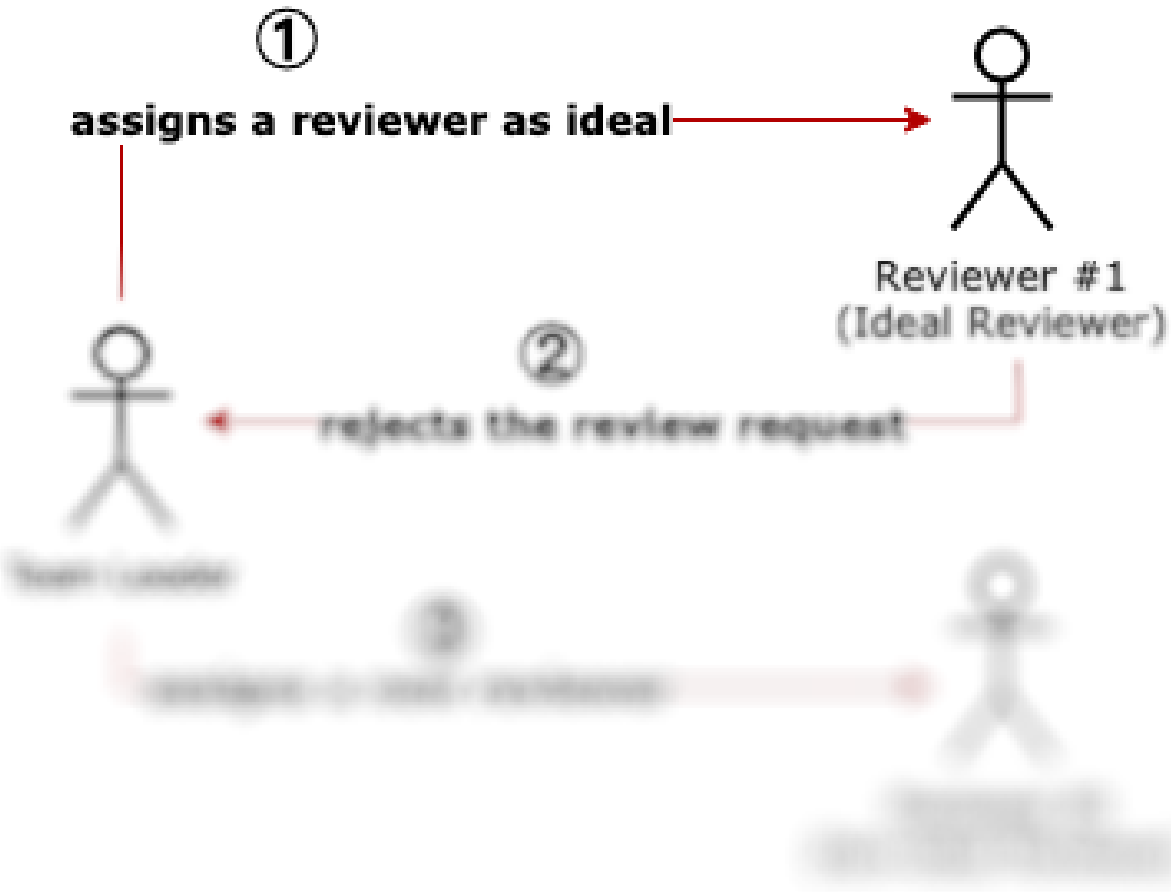


## Ground Truth in Code Reviewer Recommendation Studies:

- Recommendation models rely on the real-life assignments.
- These assignments are assumed to be ideal.
- Studies in real-life projects show that code reviewers are not usually assigned with the aim of finding the ideal one.

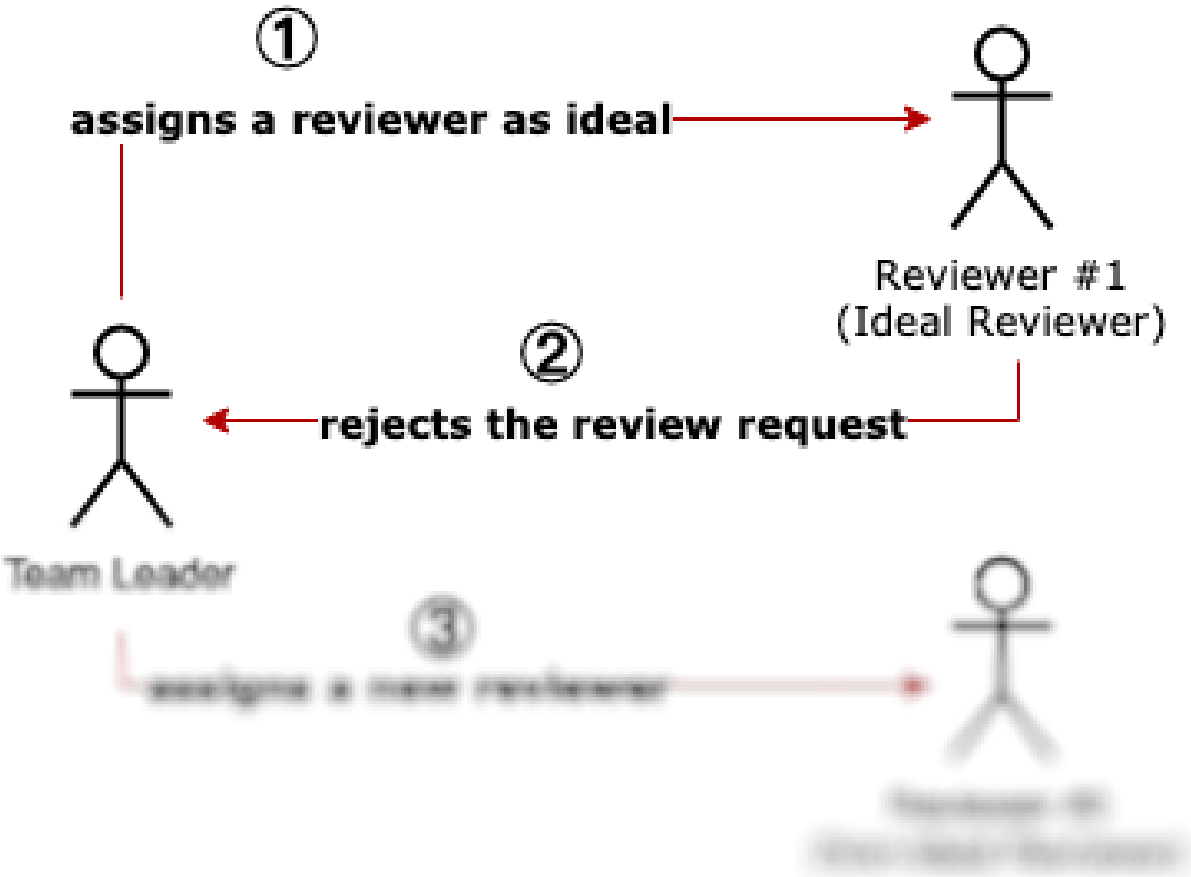
# Problematical Reviewer Selection Scenario





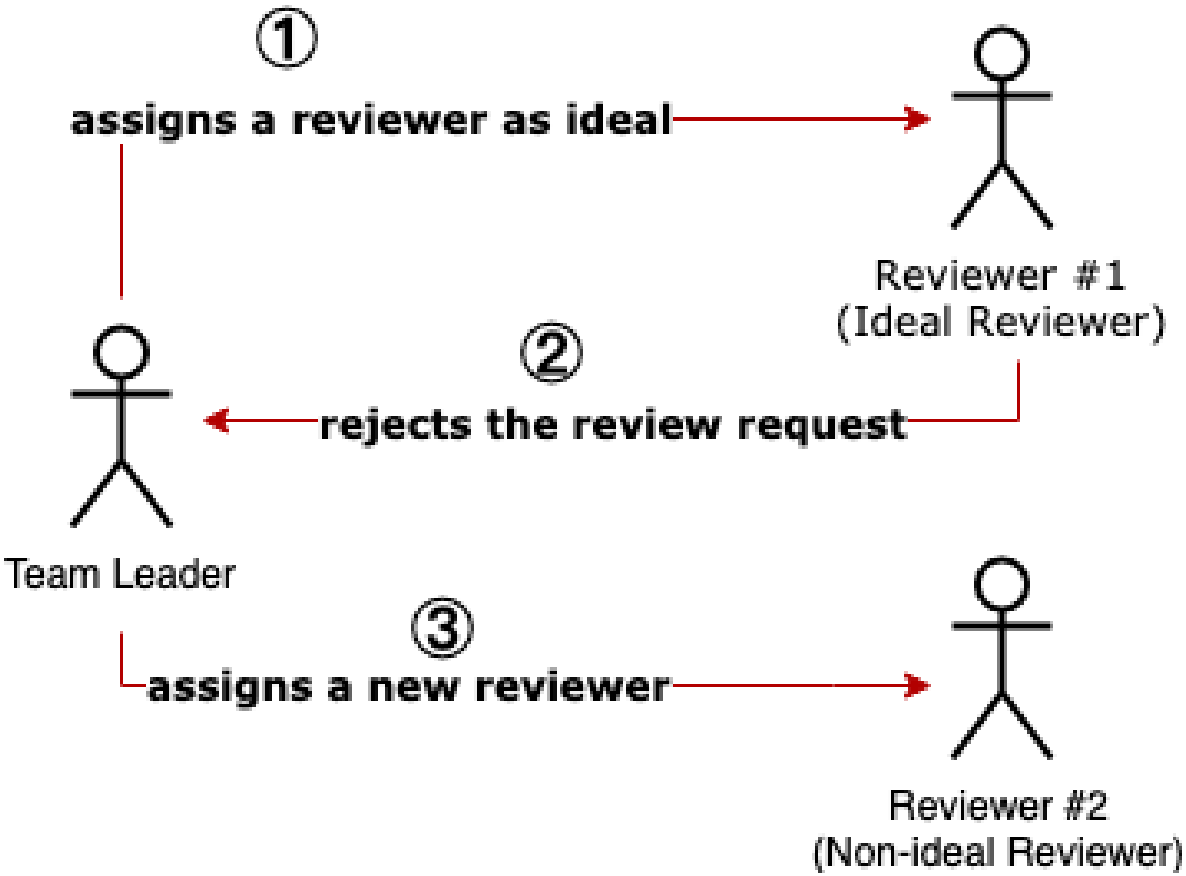
# Problematical Reviewer Selection Scenario





# Problematical Reviewer Selection Scenario





# Who is an ideal reviewer?



## Ideal Reviewer:

The theoretical best possible reviewer in the team that would improve or preferably perfect (such as pointing out all the defects) the pull request under review.

## Warning:

- In our case, the selection of ideal reviewer is assumed to be completed by only technical factors.
- i.e. If a developer is considered as the ideal reviewer for a pull request but is not available for a review at that moment, he/she is still the ideal reviewer.

# Who is an ideal reviewer?



## Ideal Reviewer:

The theoretical best possible reviewer in the team that would improve or preferably perfect (such as pointing out all the defects) the pull request under review.

## Warning:

- In our case, the selection of ideal reviewer is assumed to be completed by only technical factors.
- i.e. If a developer is considered as the ideal reviewer for a pull request but is not available for a review at that moment, he/she is still the ideal reviewer.

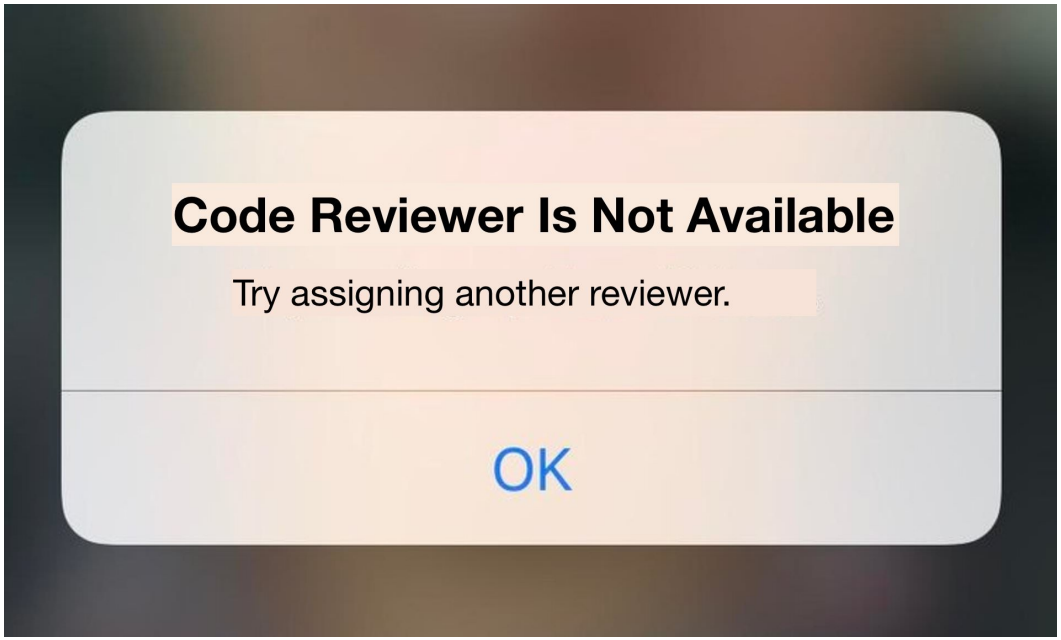
# What causes a non-ideal reviewer assignment?



**Availability Reasons:**

The ideal reviewer might be...

- physically absent from work, so he/she cannot review the pull request.
- busy with some other tasks, so he/she declines to review the pull request.
- busy with some other tasks and is late to reply the review request.



# What causes a non-ideal reviewer assignment?



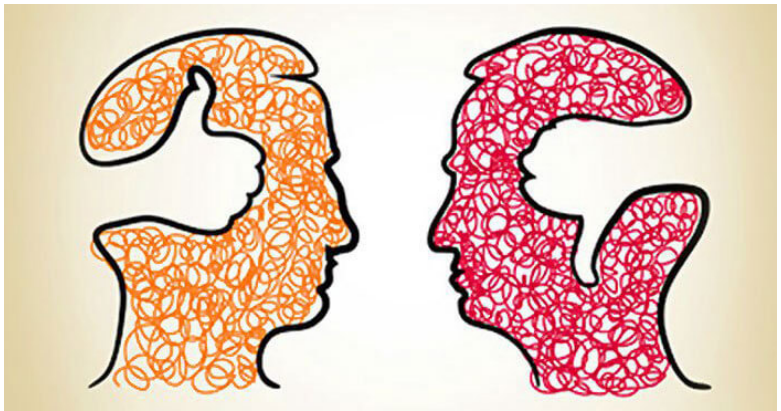
## Cognitive Bias:

### Attribute Substitution:

It occurs when an individual has to make a judgment (of a target attribute) that is computationally complex, and instead substitutes a more easily calculated heuristic attribute.

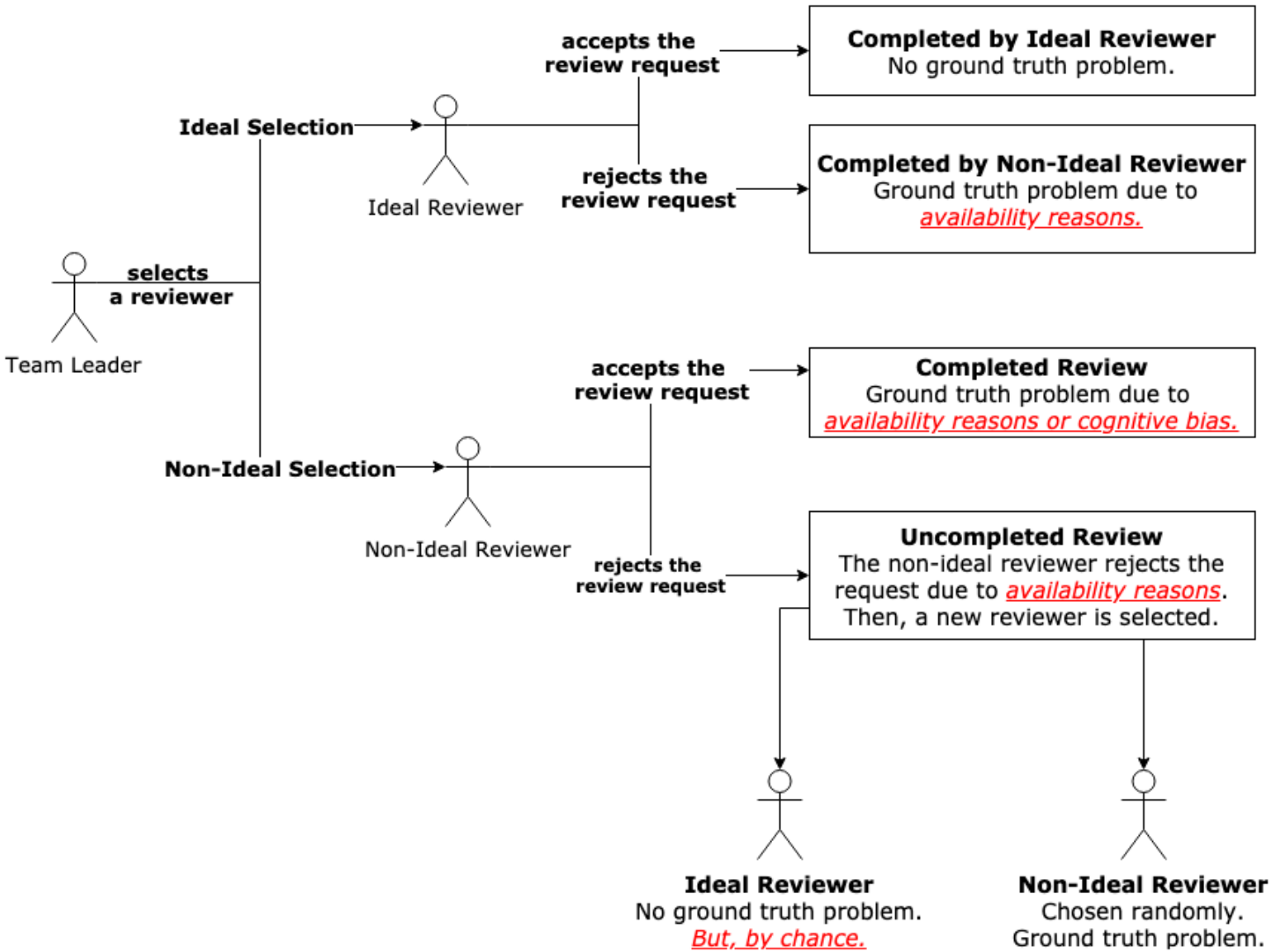
The team leader prefers to assign...

- a volunteer for the review.
- a reviewer based on their work schedule.
- a new hire as a reviewer for educational purposes.
- a developer based on their relative response time to review requests.



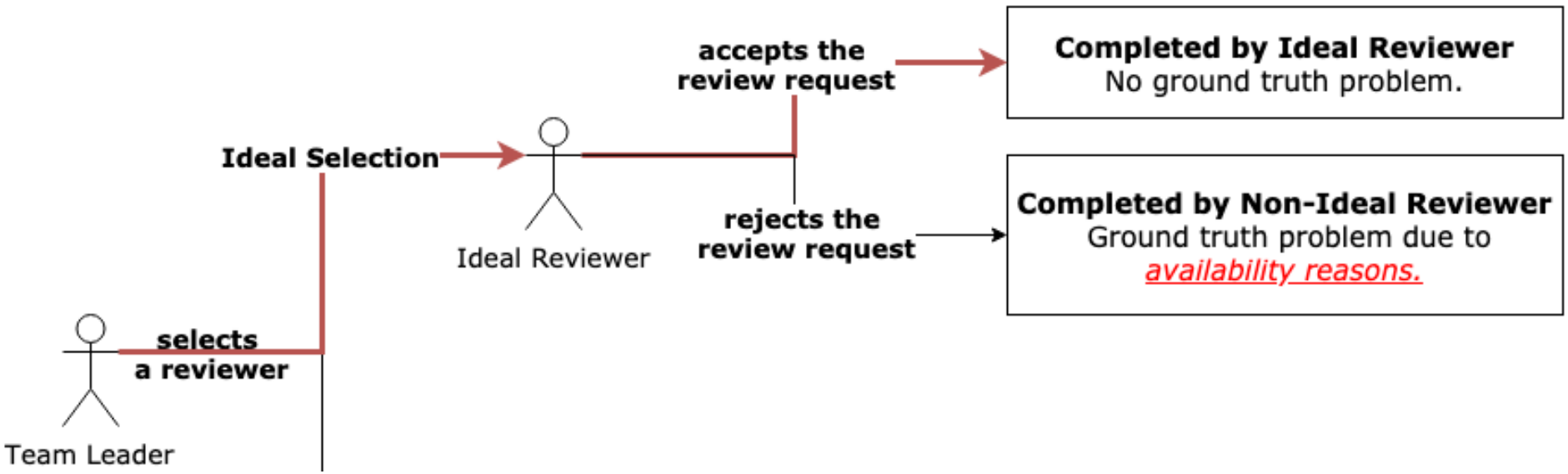
# Possible Reviewer Assignment Scenarios





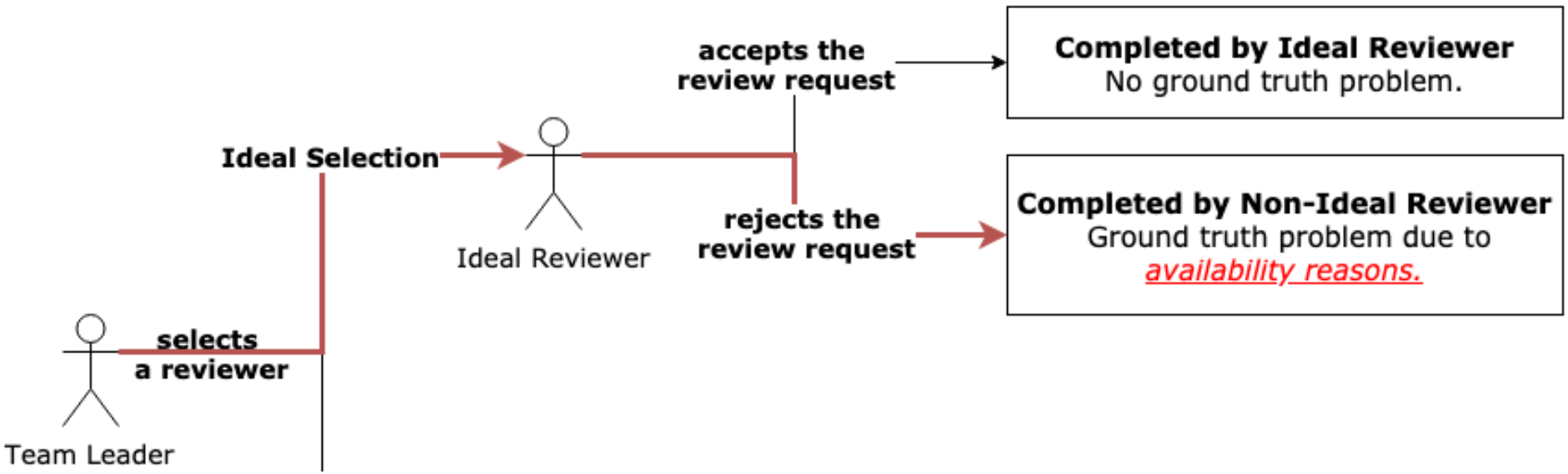
# Scenario 1




accepts the
review request
Completed by Ideal Reviewer
No ground truth problem.
Ideal Selection
Ideal Reviewer
rejects the
review request
Completed by Non-Ideal Reviewer
Ground truth problem due to
availability reasons.
selects
a reviewer
Team Leader

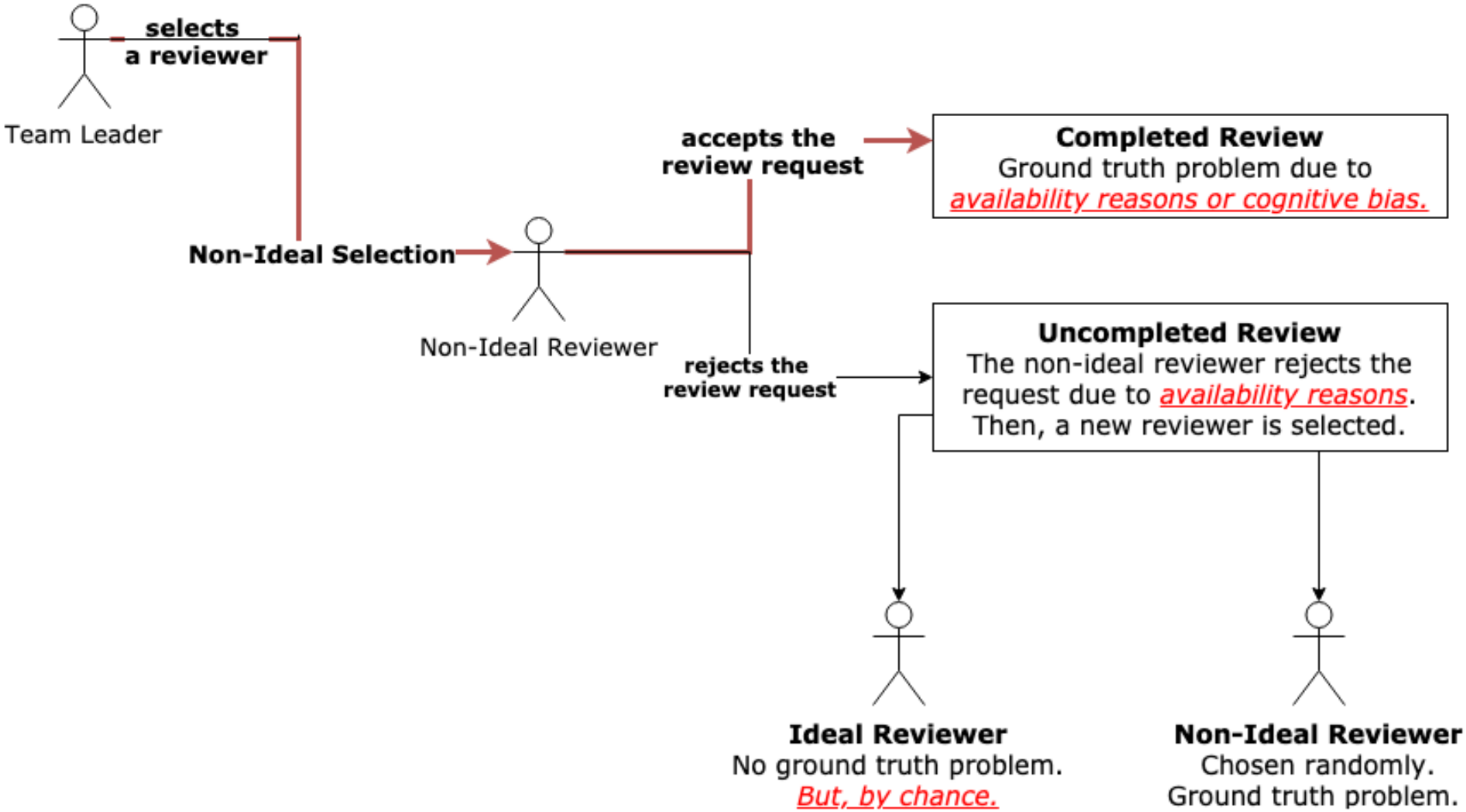
# Scenario 2




accepts the
review request
Completed by Ideal Reviewer
No ground truth problem.
Ideal Selection
Ideal Reviewer
rejects the
review request
Completed by Non-Ideal Reviewer
Ground truth problem due to
availability reasons.
selects
a reviewer
Team Leader

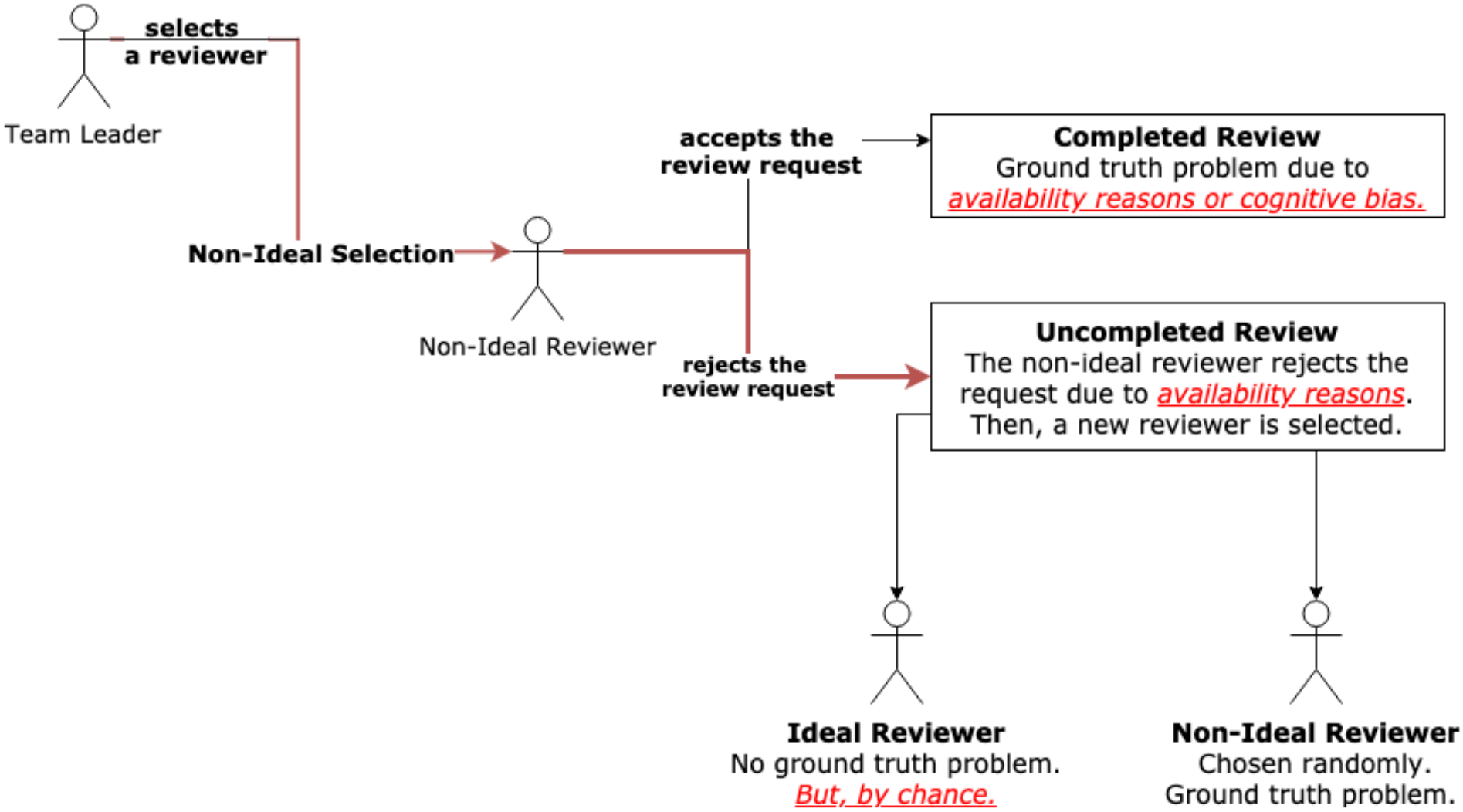
# Scenario 3




selects
a reviewer
Team Leader
Non-Ideal Selection
Non-Ideal Reviewer
accepts the
review request
Completed Review
Ground truth problem due to
availability reasons or cognitive bias.
rejects the
review request
Uncompleted Review
The non-ideal reviewer rejects the
request due to availability reasons.
Then, a new reviewer is selected.
Ideal Reviewer
No ground truth problem.
But, by chance.
Non-Ideal Reviewer
Chosen randomly.
Ground truth problem.

# Scenario 4




selects
a reviewer
Team Leader
Non-Ideal Selection
Non-Ideal Reviewer
accepts the
review request
Completed Review
Ground truth problem due to
availability reasons or cognitive bias.
rejects the
review request
Uncompleted Review
The non-ideal reviewer rejects the
request due to availability reasons.
Then, a new reviewer is selected.
Ideal Reviewer
No ground truth problem.
But, by chance.
Non-Ideal Reviewer
Chosen randomly.
Ground truth problem.

# Data, data, data...



Without data, you're just another person with an opinion.

*Dr. W. Edwards Deming*

# Quantitative Evidence from Real Life



| Project Name | Total Number of Pull Requests | Number of PRs with at least one non-responsive reviewer | The ratio of PRs having at least one non-responsive reviewer |
|---|---|---|---|
| Android | 36,771 | 24,367 | 66% |
| LibreOffice | 18,716 | 3,039 | 16% |
| Open Stack | 108,788 | 24,589 | 23% |
| Qt | 65,815 | 30,630 | 47% |
| TOTAL | 230,090 | 82,625 | 36% |

Table: An Analysis of Pull Request Reviews from 4 Large OSS Projects[1]

**Notice**

The results illustrate that 36% of pull requests suffer from the *availability reasons*.

[1] S. Ruangwan, P. Thongtanunam, A. Ihara, and K. Matsumoto, "The impact of human factors on the participation decision of reviewers in modern code review," *Empirical Software Engineering*, vol. 24, no. 2,pp. 973-1016, 2019. pp. 973–1016, 2019.

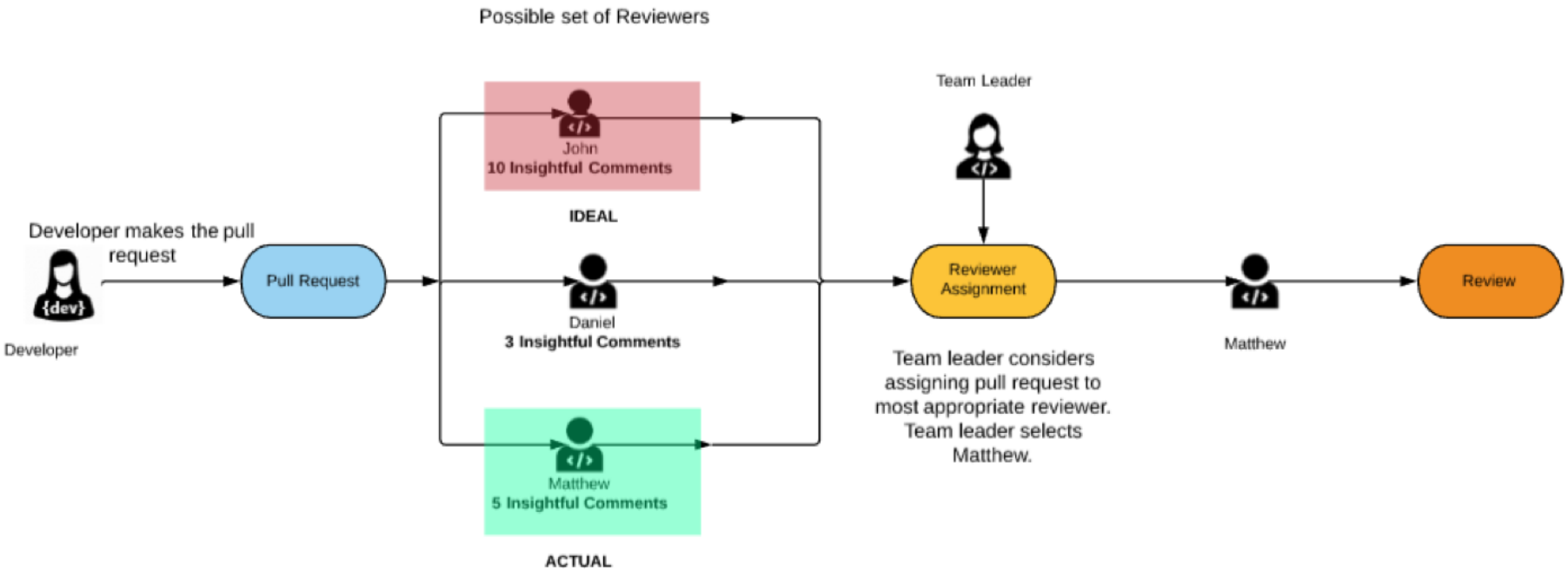
# Solution Alternatives



- Expensive Setup in Real Life
- Forward-Looking Mining

# Expensive Setup in Real Life

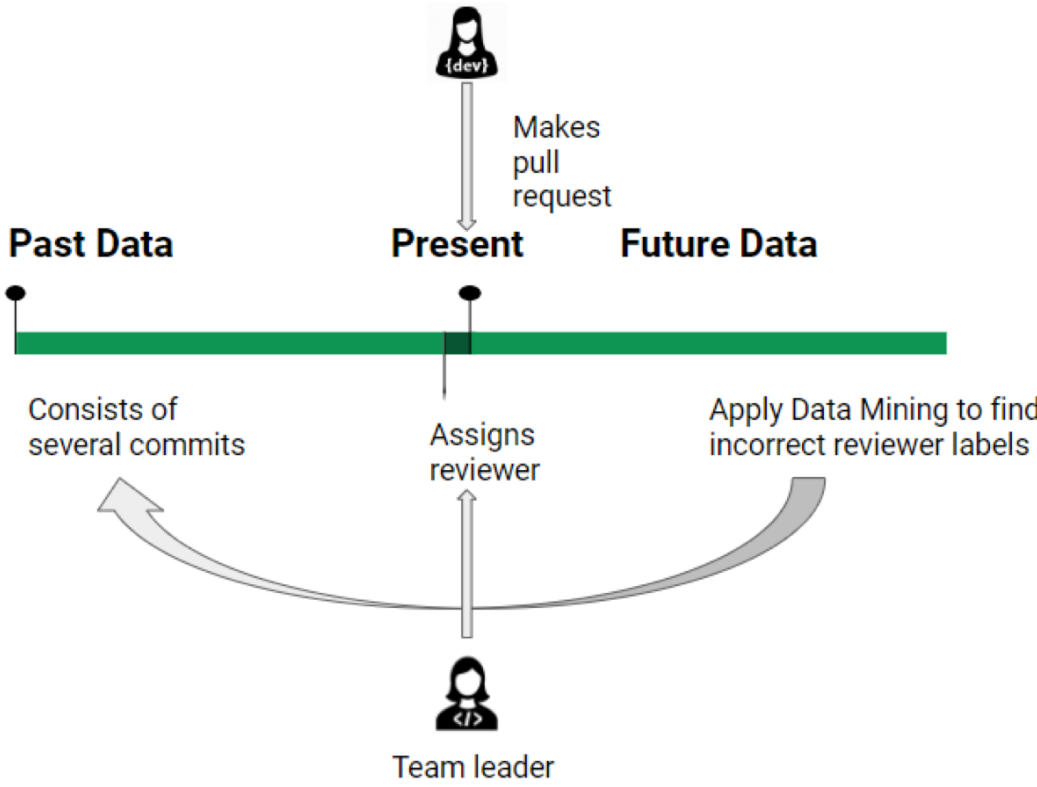




# Forward-Looking Mining



- If a bug is reopened, it is a potential indicator that the assigned reviewer was not the ideal reviewer for that pull request.
- Deleting these instances will increase the validity of the dataset.



# Conclusion



- The validation of real-life collected datasets are problematic.
- This problem is valid for other software engineering tasks. (i.e. bug localization, developer recommendation etc.)
- As future work, we are planning to introduce quantitative evidence for cognitive bias and explore alternative solutions for this problem.

# Future Work



- Introducing quantitative evidence for attribute substitution bias.
- Explore alternative solutions for this problem.

# Any Questions



For any further, please contact me.

Emre Doğan
*MSc Student*
Department of Computer Engineering
Bilkent University
Ankara, Turkey
emre.dogan@bilkent.edu.tr

# References I