Speculative Analysis for Comment Quality Assessment

Pooja Rani

Advisor: Prof. Oscar Nierstrasz

Software Composition Group

University of Bern, Switzerland

Motivation

```
/**
*A class representing a window on the screen.
*For example:
*<pre>
*Window win = new Window(parent);
*win.show();
*</pre>
*
*@author Sami Shaio
*@version 1.13, 06/08/06
*@see java.awt.BaseWindow
*/

class Window extends BaseWindow{
..
}
```

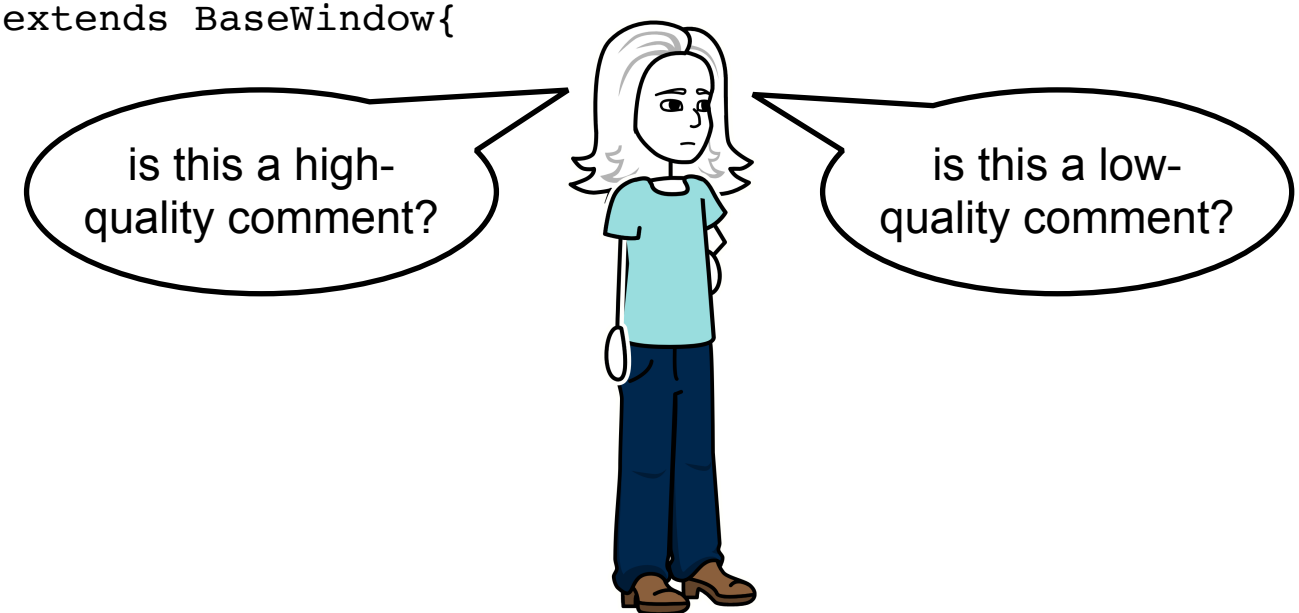
Trustworthy form of documentation

High-quality code comments assist developers

Problem

```
/**
*A class representing a window on the screen.
*For example:
*<pre>
*Window win = new Window(parent);
*win.show();
*</pre>
*
*@author Sami Shaio
*@version 1.13, 06/08/06
*@see java.awt.BaseWindow
*/

class Window extends BaseWindow{
..
}
```



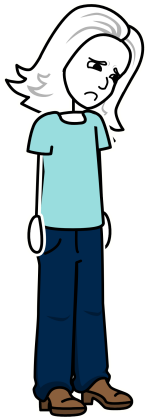
Challenges

No standard definition of comment quality

No strict syntax and structure conventions

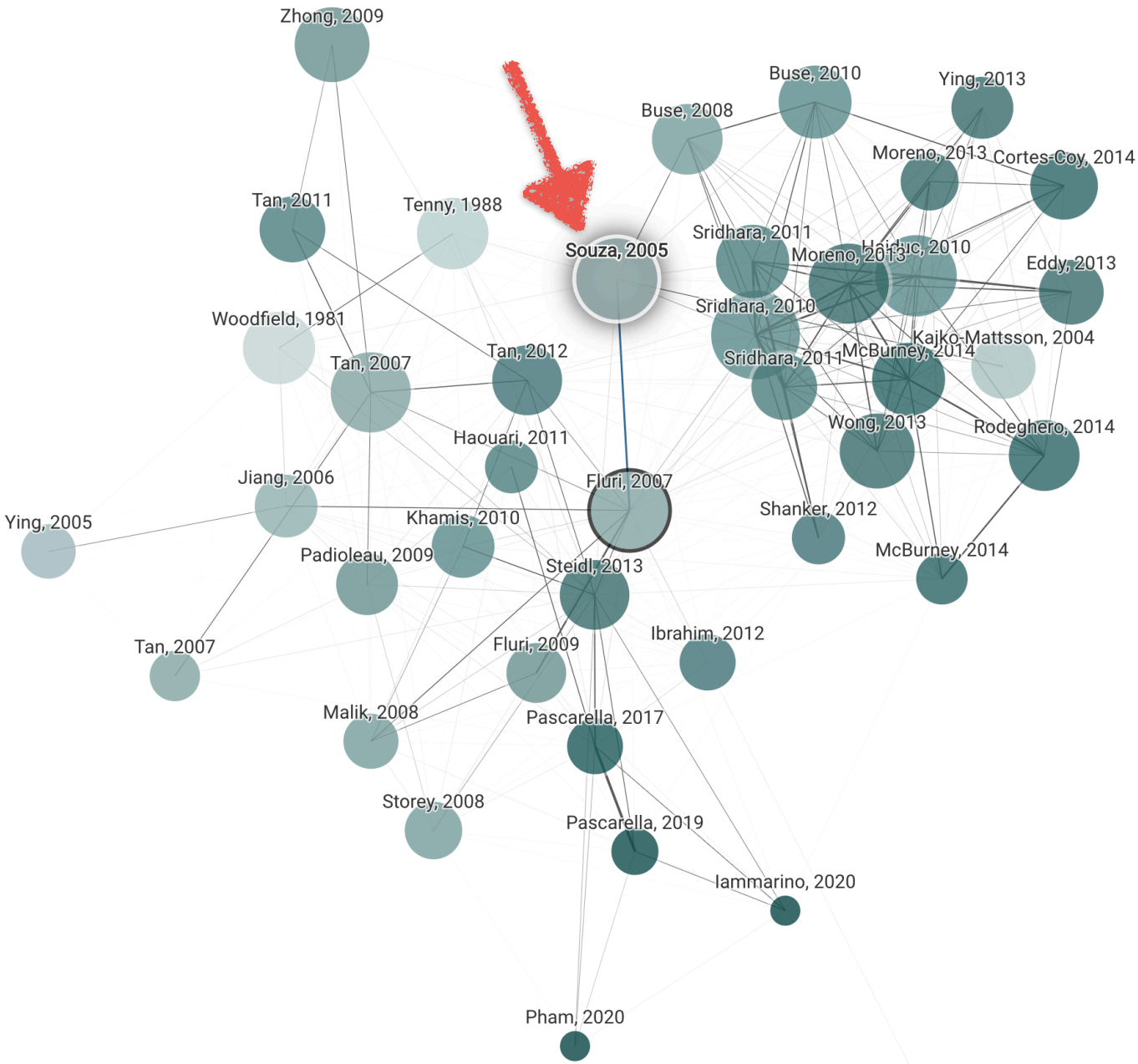
The compiler does not check comments

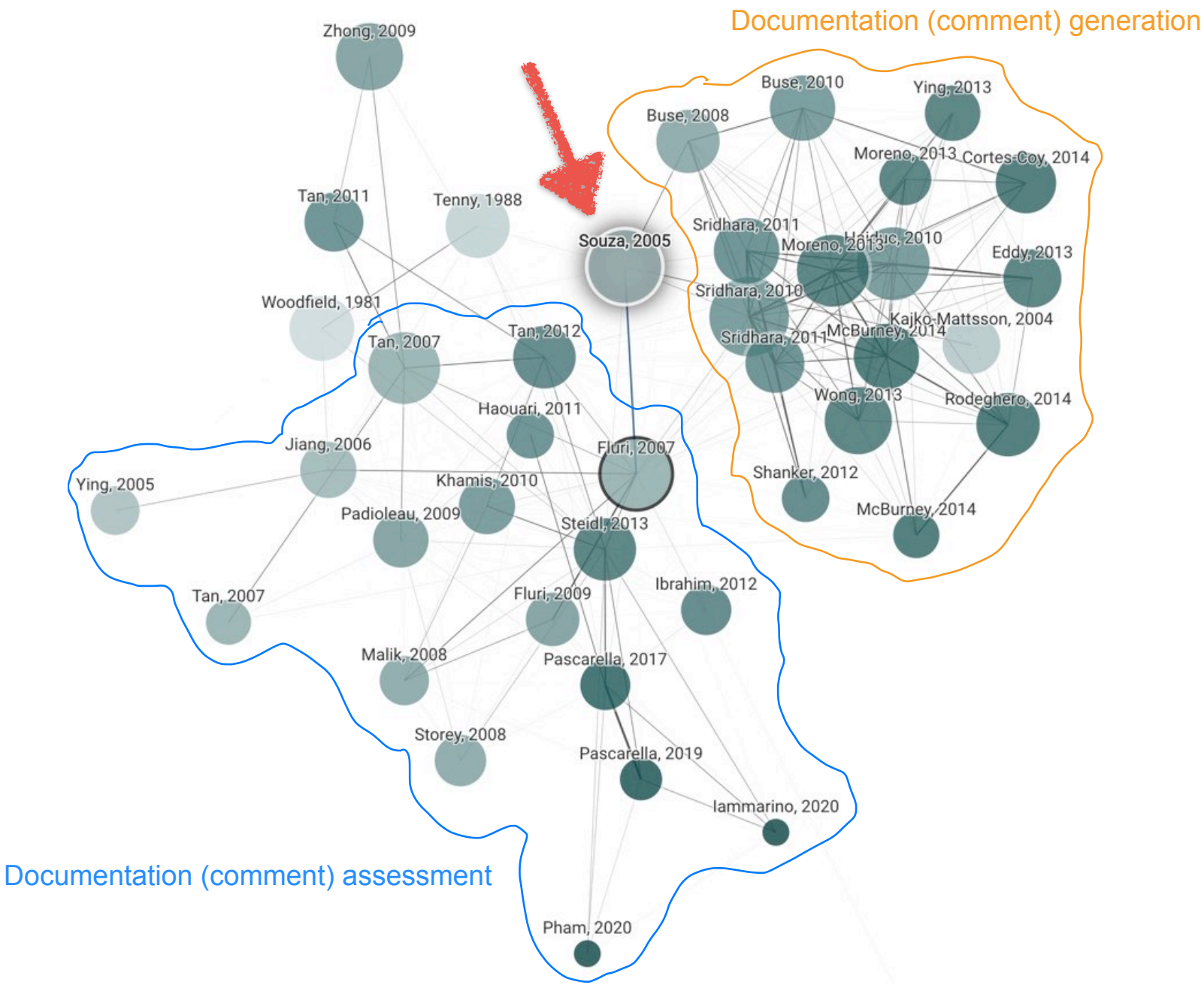
Lack of quality assessment tools



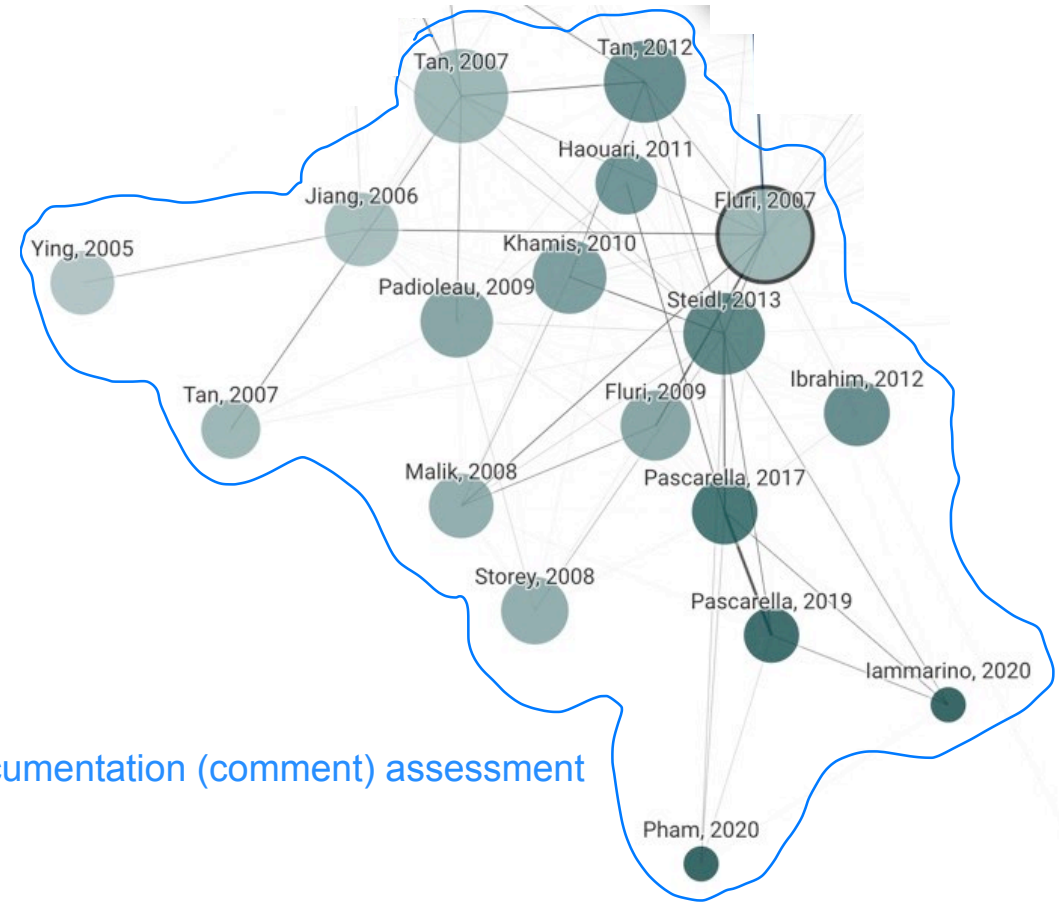
Makes quality assessment a non-trivial problem

State of the art


Zhong, 2009
Buse, 2010
Ying, 2013
Buse, 2008
Moreno, 2013
Cortes-Coy, 2014
Tan, 2011
Tenny, 1988
Souza, 2005
Sridhara, 2011
Moreno, 2013
Eddy, 2013
Sridhara, 2010
Woodfield, 1981
Kajko-Mattsson, 2004
Tan, 2012
Tan, 2007
Sridhara, 2011
McBurney, 2014
Wong, 2013
Rodeghero, 2014
Haouari, 2011
Jiang, 2006
Fluri, 2007
Shanker, 2012
Ying, 2005
Khamis, 2010
Padioleau, 2009
McBurney, 2014
Steidl, 2013
Ibrahim, 2012
Tan, 2007
Fluri, 2009
Malik, 2008
Pascarella, 2017
Storey, 2008
Pascarella, 2019
Iammarino, 2020
Pham, 2020


Documentation (comment) generation
Documentation (comment) assessment
Zhong, 2009
Tan, 2011
Tenny, 1988
Souza, 2005
Woodfield, 1981
Tan, 2007
Tan, 2012
Haouari, 2011
Jiang, 2006
Fluri, 2007
Ying, 2005
Khamis, 2010
Padioleau, 2009
Steidl, 2013
Tan, 2007
Fluri, 2009
Ibrahim, 2012
Malik, 2008
Pascarella, 2017
Storey, 2008
Pascarella, 2019
Iammarino, 2020
Pham, 2020
Buse, 2010
Ying, 2013
Buse, 2008
Moreno, 2013
Cortes-Coy, 2014
Sridhara, 2011
Moreno, 2013
Haiduc, 2010
Eddy, 2013
Sridhara, 2010
Kajko-Mattsson, 2004
Sridhara, 2011
McBurney, 2014
Wong, 2013
Rodeghero, 2014
Shanker, 2012
McBurney, 2014

Not all of them focused mainly on comments or all aspects of comments



Documentation (comment) assessment

Given the increasing use of multi-language environment, we need a deeper understanding of developer commenting practices and concerns to achieve high-quality comments

Thesis statement

“Understanding the specification of high-quality comments to build effective assessment tools requires a multi-perspective view of the comments”.

Thesis statement

“Understanding the specification of high-quality comments to build effective assessment tools requires a multi-perspective view of the comments. The view can be approached by”

RQ1: What do developers write in comments across languages?

Thesis statement

“Understanding the specification of high-quality comments to build effective assessment tools requires a multi-perspective view of the comments. The view can be approached by”

RQ1: What do developers write in comments across languages?

RQ2: What do developers ask about code commenting practices?

Thesis statement

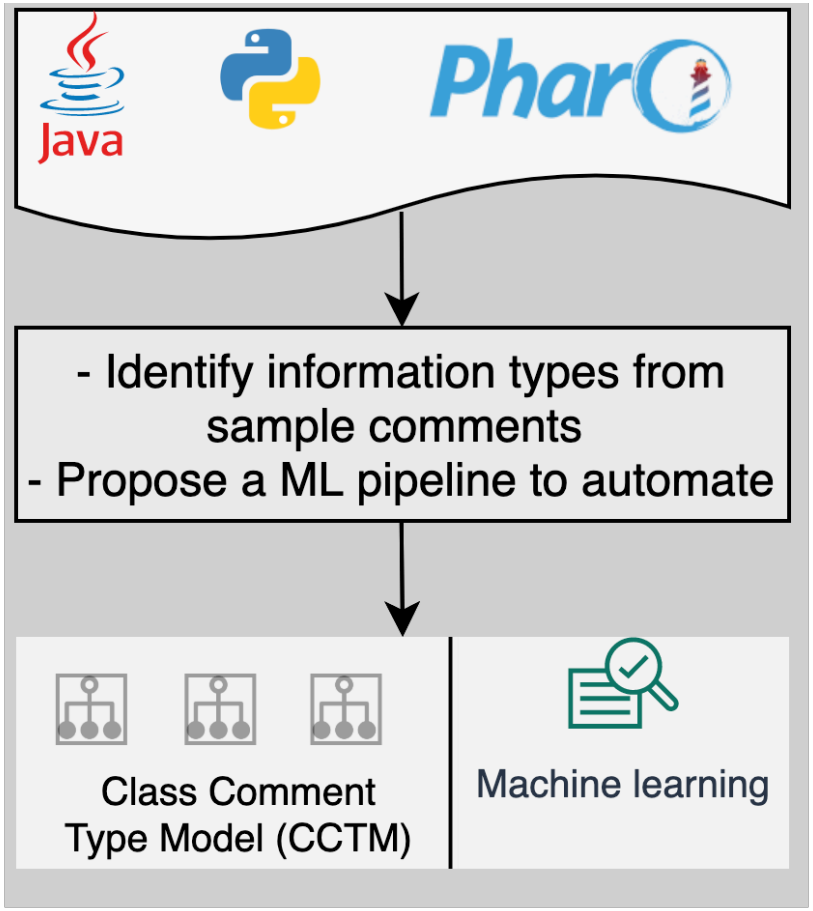
"Understanding the specification of high-quality comments to build effective assessment tools requires a multi-perspective view of the comments. The view can be approached by"

RQ1: What do developers write in comments across languages?

RQ2: What do developers ask about code commenting practices?

RQ3: What quality attributes are often considered in assessing comment quality?

RQ1: What do developers write in comments across languages?



Java, Python, Smalltalk,
20 projects

Extract class comments,
37, 446 comments

Classify sample comments,
1, 066 comments

Output: a taxonomy, and a classifier

RQ1: What do developers write in comments across languages?

```
/**
*A class representing a window on the screen.
*For example:
*<pre>
*Window win = new Window(parent);
*win.show();
*</pre>
*
*@author Sami Shaio
*@version 1.13, 06/08/06
*@see java.awt.BaseWindow
*/

class Window extends BaseWindow{
..
}
```

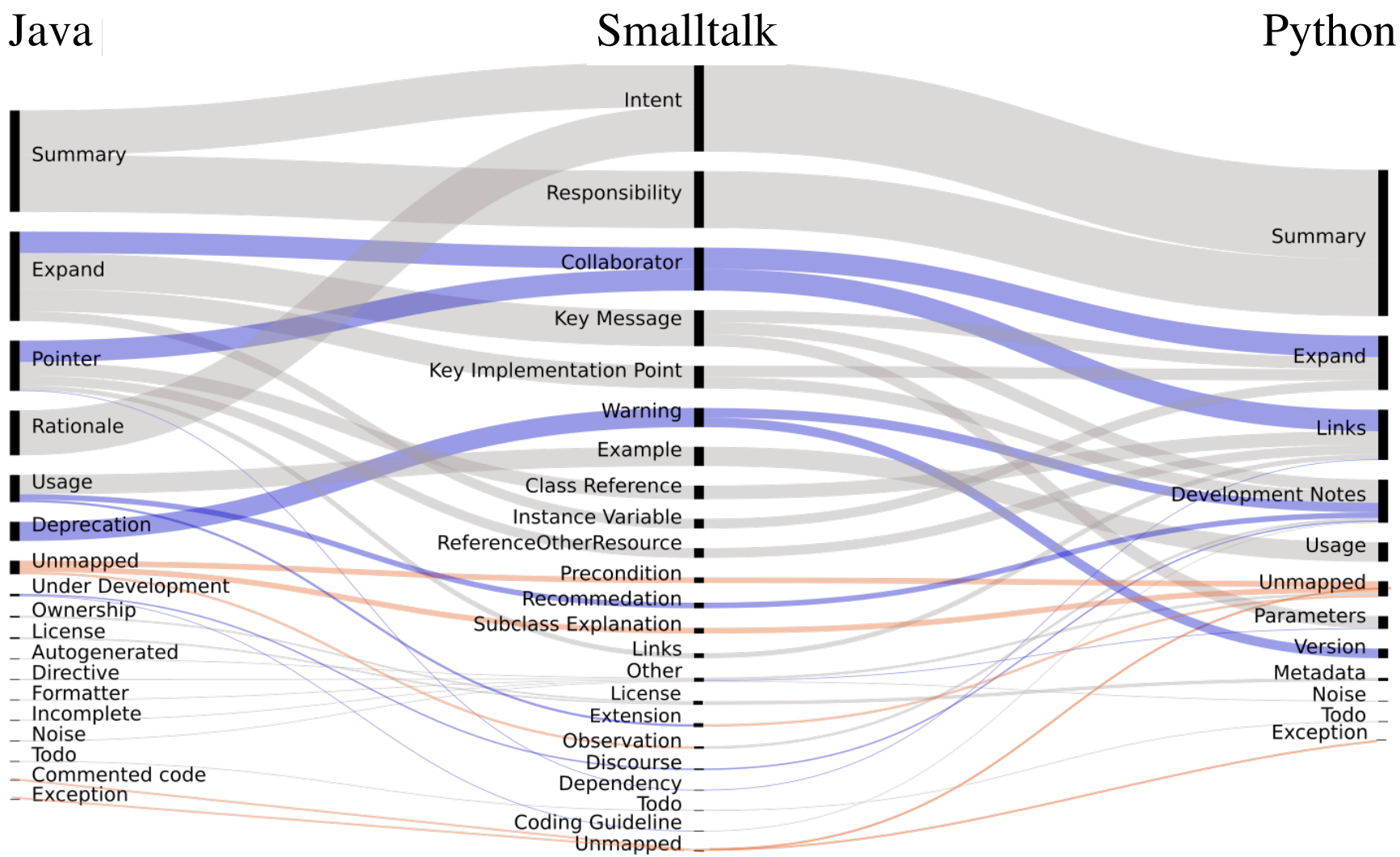
Summary

Usage

Ownership

Pointer

RQ1: What do developers write in comments across languages?



CCTM

Automatically identify an information type

```
/**
*A class representing a window on the screen.
*For example:
*<pre>
*Window win = new Window(parent);
*win.show();
*</pre>
*
*@author Sami Shaio
*@version 1.13, 06/08/06
*@see java.awt.BaseWindow
*/

class Window extends BaseWindow{
..
}
```

Summary

Pointer

Recurrent natural language patterns in writing a specific type of information

Automatically identify an information type

```
/**
*A class representing a window on the screen.
*For example:
*<pre>
*Window win = new Window(parent);
*win.show();
*</pre>
*
*@author Sami Shaio
*@version 1.13, 06/08/06
*@see java.awt.BaseWindow
*/

class Window extends BaseWindow{
..
}
```

Summary

Represents [something]

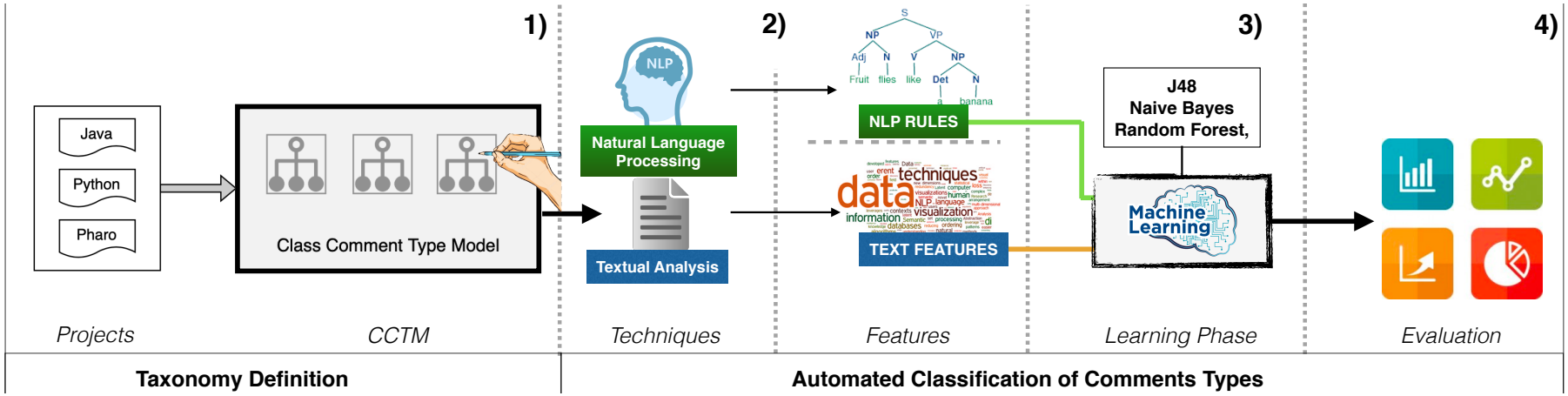
[verb]s [noun]

Pointer

Sees [something]

Recurrent natural language patterns in writing a specific type of information

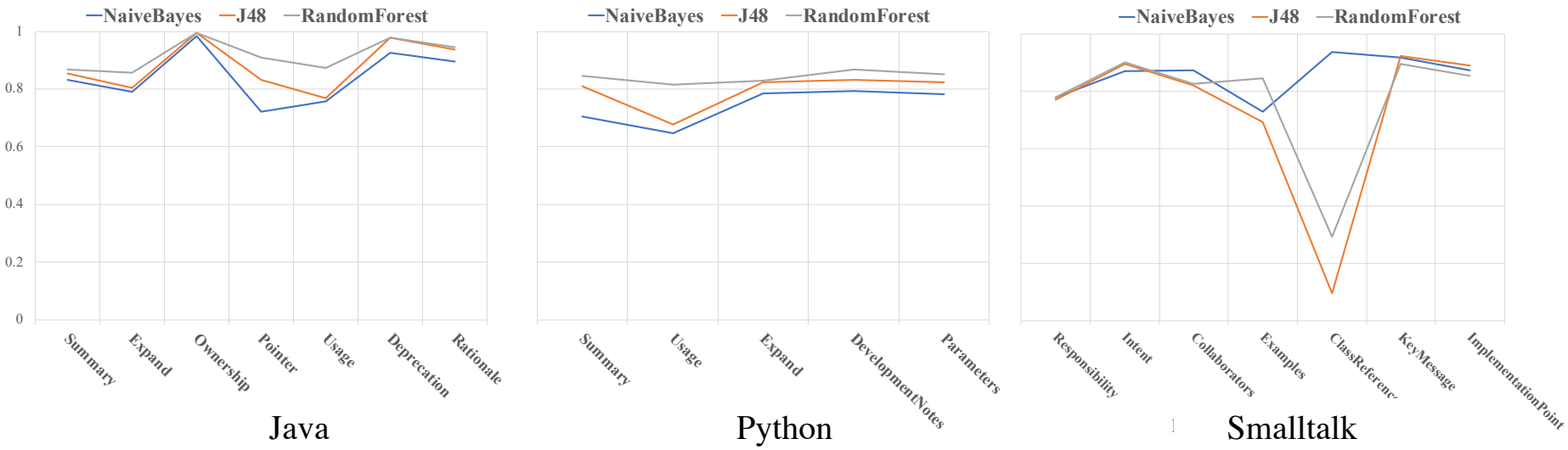
Automatically identify an information type



Ground truth: manually analyzed comments

Used recurrent patterns as feature set + text features

Automatically identify an information type



Random Forest technique classifies comments better

Takeaway

Developers embed at least 10 types of information in comments across languages

Using recurrent natural language patterns as features improves the classification

We are currently testing our classification approach using the deep learning technique (FastText)

Timeline & Contribution

What do class comments tell us? An investigation of comment evolution and practices in Pharo Smalltalk

An investigation of comment evolution, content and their adherence to style guideline

How to Identify Class Comment Types? A Multi-language Approach for Class Comments Classification

An investigation of comments in Java, Python, Smalltalk, and develop an approach to automatically identify information types across languages

In the stage of acceptance

What do class comments tell us? An investigation of comment evolution and practices in Pharo Smalltalk

Pooja Rani · Sebastiano Panichella · Manuel Leuenberger · Mohammad Ghafari · Oscar Nierstrasz

Received: date / Accepted: date

Abstract Previous studies have characterized code comments in various programming languages, and have shown how a high quality of code comments is crucial to support program comprehension activities and to improve the effectiveness of maintenance tasks. However, very few studies have focused on the analysis of the information embedded in code comments. None of them has compared developer practices to w
standard guidelines or analyzed these characteristics in the Pha

These class commenting practices have their origins in S
years. Smalltalk traditionally separates class comments from s
template for entering a comment for newly-created classes. T
over the years, particularly in the Pharo environment. This pap
study investigating commenting practices in Pharo Smalltalk.
class comment evolution over seven Pharo versions. Then, w
tively analyze class comments of the most recent version of Ph
mation types of Pharo comments. Finally, we study the adheren
practices to the class template over Pharo versions.

The results of this study show that there is a rapid increase
tial three Pharo versions, while in subsequent versions develop
new and old classes, thus maintaining a similar ratio. In additio
tics of the comments from the latest Pharo version suggests t
typically embedded in class comments by developers and that o
in the latest *Pharo class comment template*. However, the infor
standard template tend to be present more often than other typ
ally, we find that a substantial proportion of comments follow
plate in writing these information types, but they are written an
way. This suggests the need to standardize the commenting g

Accepted

Empirical Software Engineering (EMSE), 2021

How to Identify Class Comment Types?
A Multi-language Approach for Class Comments Classification

Pooja Rani[a], Sebastiano Panichella[b], Manuel Leuenberger[a], Andrea Di Sorbo[c], Oscar Nierstrasz[a]

[a]*Software Composition Group, University of Bern, Switzerland*
[b]*Zurich University of Applied Science, Switzerland*
[c]*Department of Engineering, University of Sannio, Italy*

Abstract

Most software maintenance and evolution tasks require developers to understand the source code of their software systems. Software developers usually inspect class comments to gain knowledge about program behavior, regardless of the programming language they are using. Unfortunately, (i) different programming languages present language-specific code commenting notations and guidelines; and (ii) the source code of software projects often lacks comments that adequately describe the class behavior, which complicates program comprehension and evolution activities. To handle these challenges, this paper investigates the different language-specific class commenting practices of three programming languages: Python, Java, and Pharo. In particular, we systematically analyze the similarities and differences of the information types found in class comments of projects developed in these languages. We propose an approach that leverages two techniques — namely Natural Language Processing and Text Analysis — to automatically identify *class comment types*, *i.e.,* the specific types of semantic information found in class comments. To the best of our knowledge, no previous work has provided a comprehensive taxonomy of class comment types for these three programming languages with the help of a common automated approach. Our results confirm that our approach can classify frequent class comment information types with high accuracy for the Python, Java, and Pharo programming languages. We believe this work can help to monitor and assess the quality and evolution of code comments in different programming languages, and thus support maintenance and evolution tasks.

Keywords: Natural Language Processing Technique, Code Comment Analysis, Software Documentation

1. Introduction

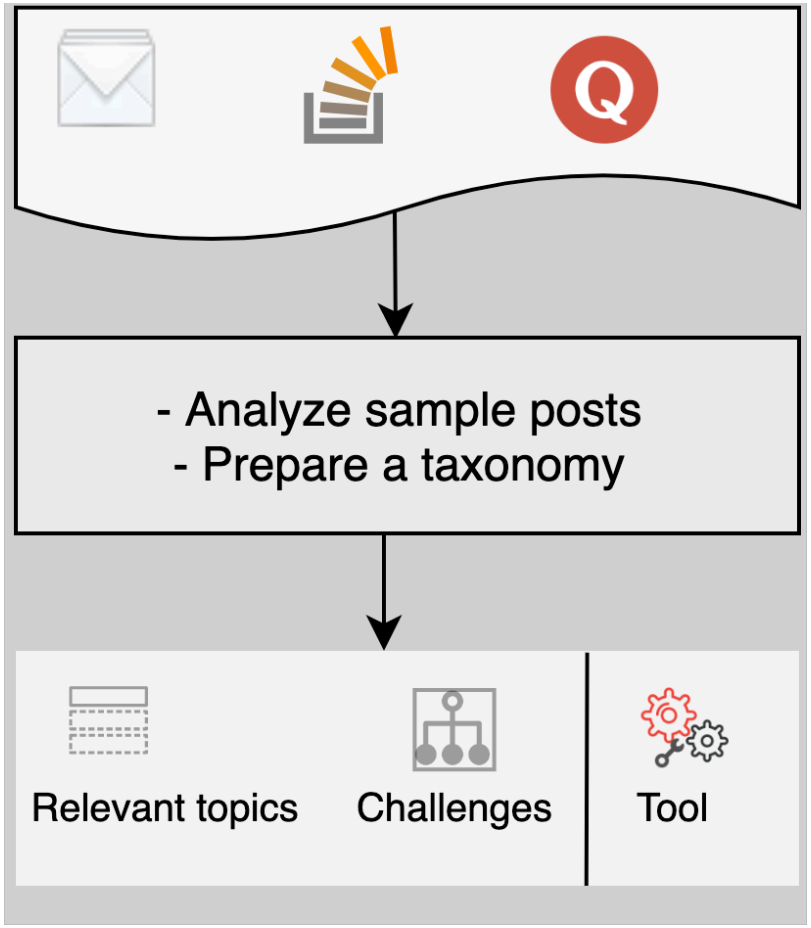
Software maintenance and evolution tasks require developers to perform program comprehension activities [1, 2]. To understand a software system, developers usually refer to the software documentation of the system [3, 4]. Previous studies have demonstrated that developers trust code comments more than other forms of documentation when they try to answer program comprehension questions [5, 6, 4]. In addition, recent work has also demonstrated that *"code documentation"* is the most used source of information for bug fixing, implementing features, communication, and even code review [7]. In particular, well-documented code simplifies software maintenance activities, but many programmers often overlook or delay code commenting tasks [8].

Under minor revision

Journal of Systems and Software (JSS), 2021

RQ2: What do developers ask about comments?

RQ2: What do developers ask about comments?



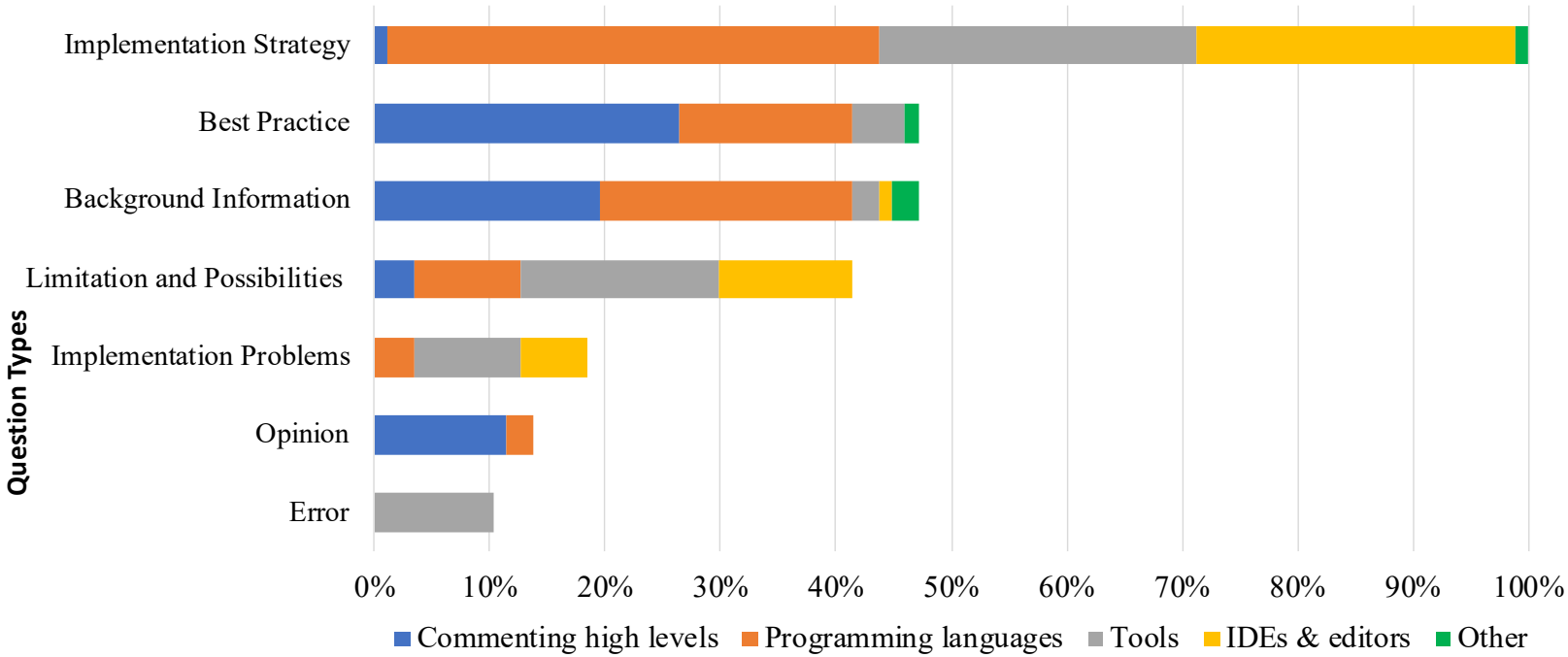
Mailing lists, Stack Overflow, Quora

Extract discussion,
23, 631 posts

Classify sample discussion,
1, 400 posts

Output: a taxonomy, guidelines, tool

RQ2: What do developers ask about comments?



Takeaway

Developers seek commenting conventions to write quality comments but often find it hard to locate them

To what extent developers follow comment conventions in writing comments?

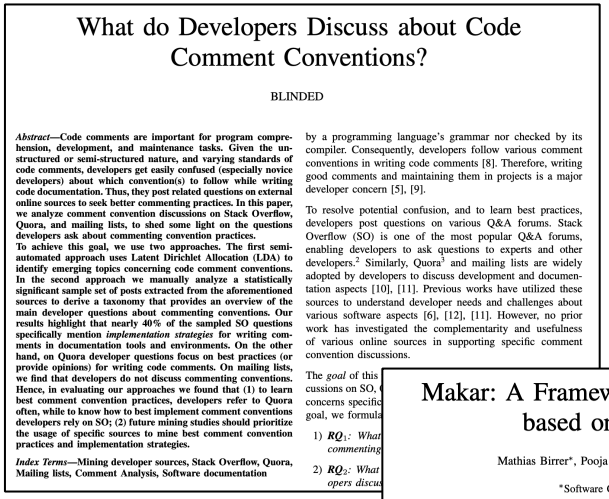
Timeline & Contribution

What do Developers Discuss about Comment Conventions?

An investigation of developer concerns related to comments on various online platforms

Makar: A Framework for Multi-source Studies based on Unstructured Data

A tool to conduct study on multiple sources such as emails, stack overflow, GitHub



What do Developers Discuss about Code Comment Conventions?

BLINDED

***Abstract*—Code comments are important for program comprehension, development, and maintenance tasks. Given the unstructured or semi-structured nature, and varying standards of code comments, developers get easily confused (especially novice developers) about which convention(s) to follow while writing code documentation. Thus, they post related questions on external online sources to seek better commenting practices. In this paper, we analyze comment convention discussions on Stack Overflow, Quora, and mailing lists, to shed some light on the questions developers ask about commenting convention practices.**
To achieve this goal, we use two approaches. The first semi-automated approach uses Latent Dirichlet Allocation (LDA) to identify emerging topics concerning code comment conventions. In the second approach we manually analyze a statistically significant sample set of posts extracted from the aforementioned sources to derive a taxonomy that provides an overview of the main developer questions about commenting conventions. Our results highlight that nearly 40% of the sampled SO questions specifically mention *implementation strategies* for writing comments in documentation tools and environments. On the other hand, on Quora developer questions focus on best practices (or provide opinions) for writing code comments. On mailing lists, we find that developers do not discuss commenting conventions. Hence, in evaluating our approaches we found that (1) to learn best comment convention practices, developers refer to Quora often, while to know how to best implement comment conventions developers rely on SO; (2) future mining studies should prioritize the usage of specific sources to mine best comment convention practices and implementation strategies.

***Index Terms*—Mining developer sources, Stack Overflow, Quora, Mailing lists, Comment Analysis, Software documentation**

by a programming language's grammar nor checked by its compiler. Consequently, developers follow various comment conventions in writing code comments [8]. Therefore, writing good comments and maintaining them in projects is a major developer concern [5], [9].

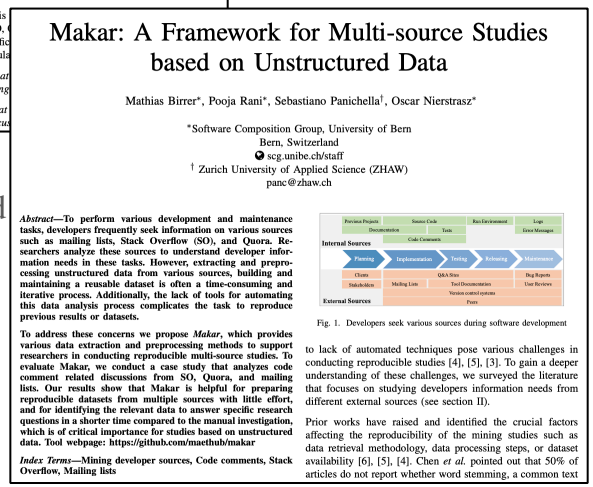
To resolve potential confusion, and to learn best practices, developers post questions on various Q&A forums. Stack Overflow (SO) is one of the most popular Q&A forums, enabling developers to ask questions to experts and other developers.[2] Similarly, Quora[3] and mailing lists are widely adopted by developers to discuss development and documentation aspects [10], [11]. Previous works have utilized these sources to understand developer needs and challenges about various software aspects [6], [12], [11]. However, no prior work has investigated the complementarity and usefulness of various online sources in supporting specific comment convention discussions.

The *goal* of this
cussions on SO,
concerns specific
goal, we formula

1) ***RQ*$_1$**: *What*
commenting
2) ***RQ*$_2$**: *What*
opers discu

In preparation

Source Code Analysis and Manipulation (SCAM), 2021



Makar: A Framework for Multi-source Studies based on Unstructured Data

Mathias Birrer*, Pooja Rani*, Sebastiano Panichella†, Oscar Nierstrasz*

*Software Composition Group, University of Bern
Bern, Switzerland
scg.unibe.ch/staff
† Zurich University of Applied Science (ZHAW)
panc@zhaw.ch

***Abstract*—To perform various development and maintenance tasks, developers frequently seek information on various sources such as mailing lists, Stack Overflow (SO), and Quora. Researchers analyze these sources to understand developer information needs in these tasks. However, extracting and preprocessing unstructured data from various sources, building and maintaining a reusable dataset is often a time-consuming and iterative process. Additionally, the lack of tools for automating this data analysis process complicates the task to reproduce previous results or datasets.**

To address these concerns we propose *Makar*, which provides various data extraction and preprocessing methods to support researchers in conducting reproducible multi-source studies. To evaluate Makar, we conduct a case study that analyzes code comment related discussions from SO, Quora, and mailing lists. Our results show that Makar is helpful for preparing reproducible datasets from multiple sources with little effort, and for identifying the relevant data to answer specific research questions in a shorter time compared to the manual investigation, which is of critical importance for studies based on unstructured data. Tool webpage: https://github.com/maethub/makar

***Index Terms*—Mining developer sources, Code comments, Stack Overflow, Mailing lists**

Fig. 1. Developers seek various sources during software development

to lack of automated techniques pose various challenges in conducting reproducible studies [4], [5], [3]. To gain a deeper understanding of these challenges, we surveyed the literature that focuses on studying developers information needs from different external sources (see section II).

Prior works have raised and identified the crucial factors affecting the reproducibility of the mining studies such as data retrieval methodology, data processing steps, or dataset availability [6], [5], [4]. Chen *et al.* pointed out that 50% of articles do not report whether word stemming, a common text

Accepted

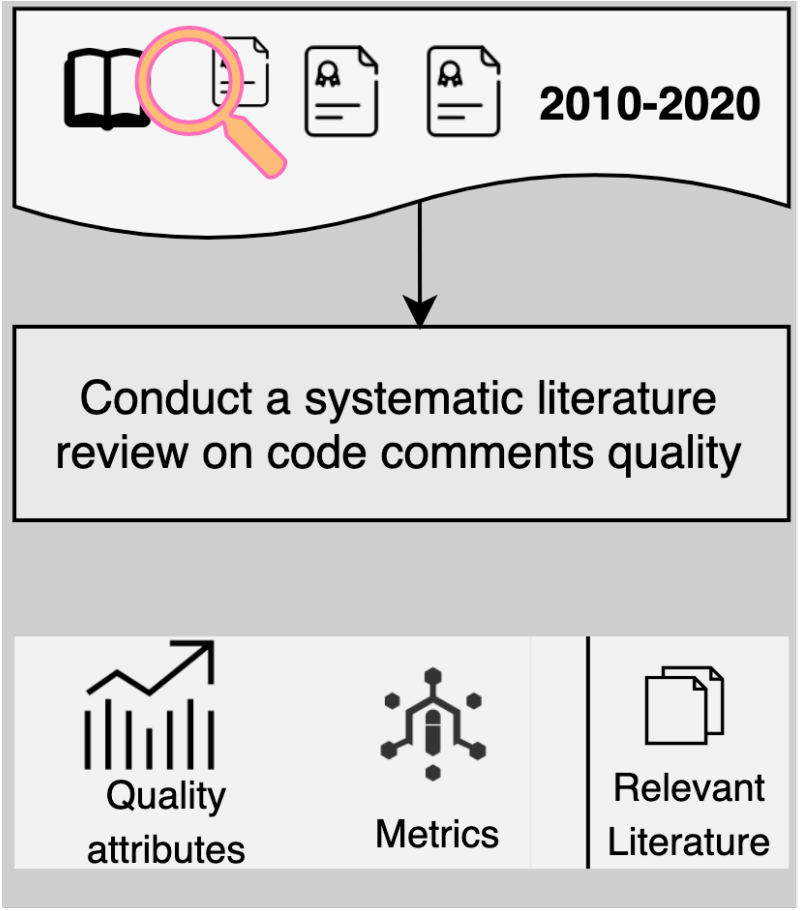
Software Analysis, Evolution and Reengineering (SANER), 2021

In the stage of acceptance

In the stage of submission

RQ3: What quality attributes are often considered in assessing comment quality?

RQ3: What quality attributes are often considered in assessing comment quality?



SE conferences & journals
195 venues

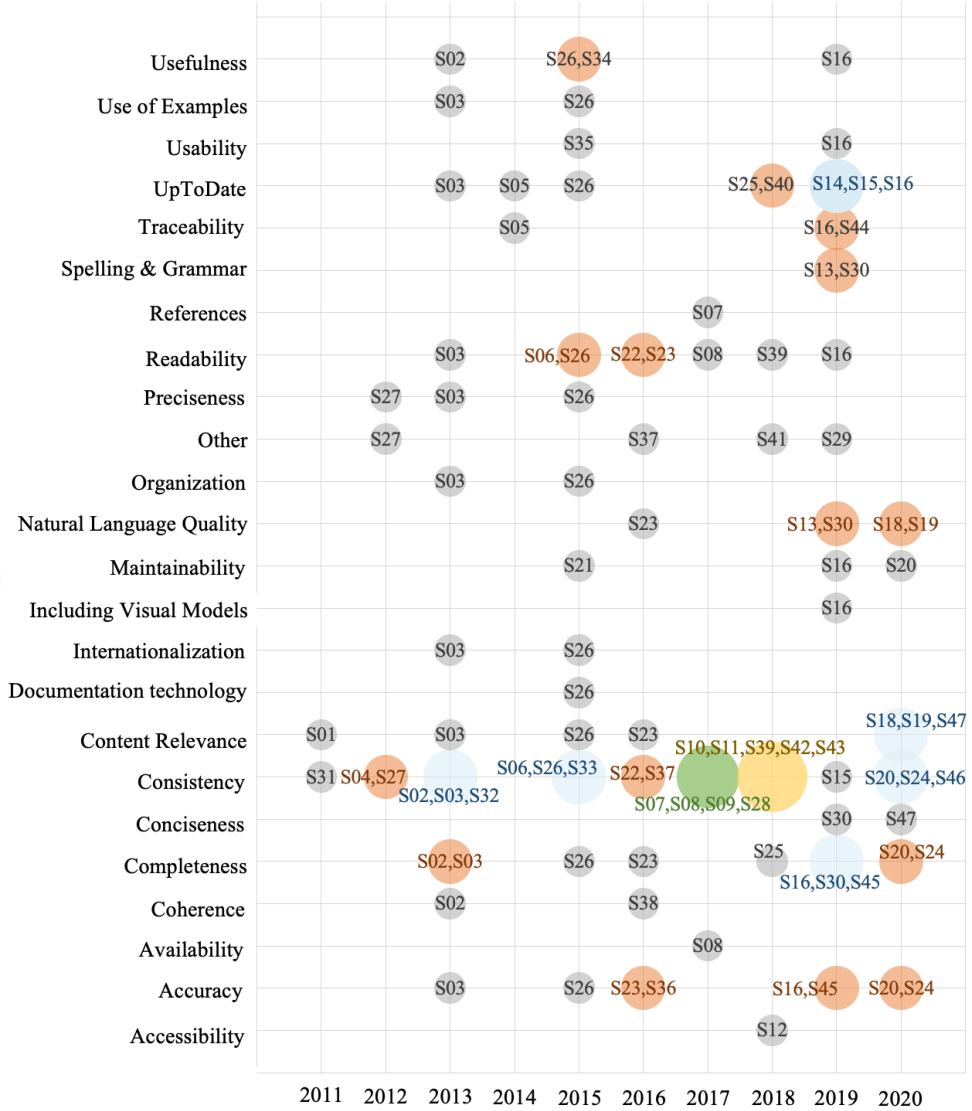
Extract proceedings,
332 proceedings

Identify candidate papers
2, 353 papers

Select relevant papers,
47 papers

Output: quality attributes, metrics

RQ3: What quality attributes are often considered in assessing comment quality?



RQ3: What quality attributes are often considered in assessing comment quality?

RQ3: What quality attributes are often considered in assessing comment quality?

Takeaway

Researchers are focusing often on handful of quality attributes in assessing comment quality

Numerous studies suggest the benefits of usability, or accessibility of comments. We need to focus on assessing comments w.r.t them.

Timeline & Contribution

A SLR about comments

An investigation of quality attributes, and techniques used for comment assessment

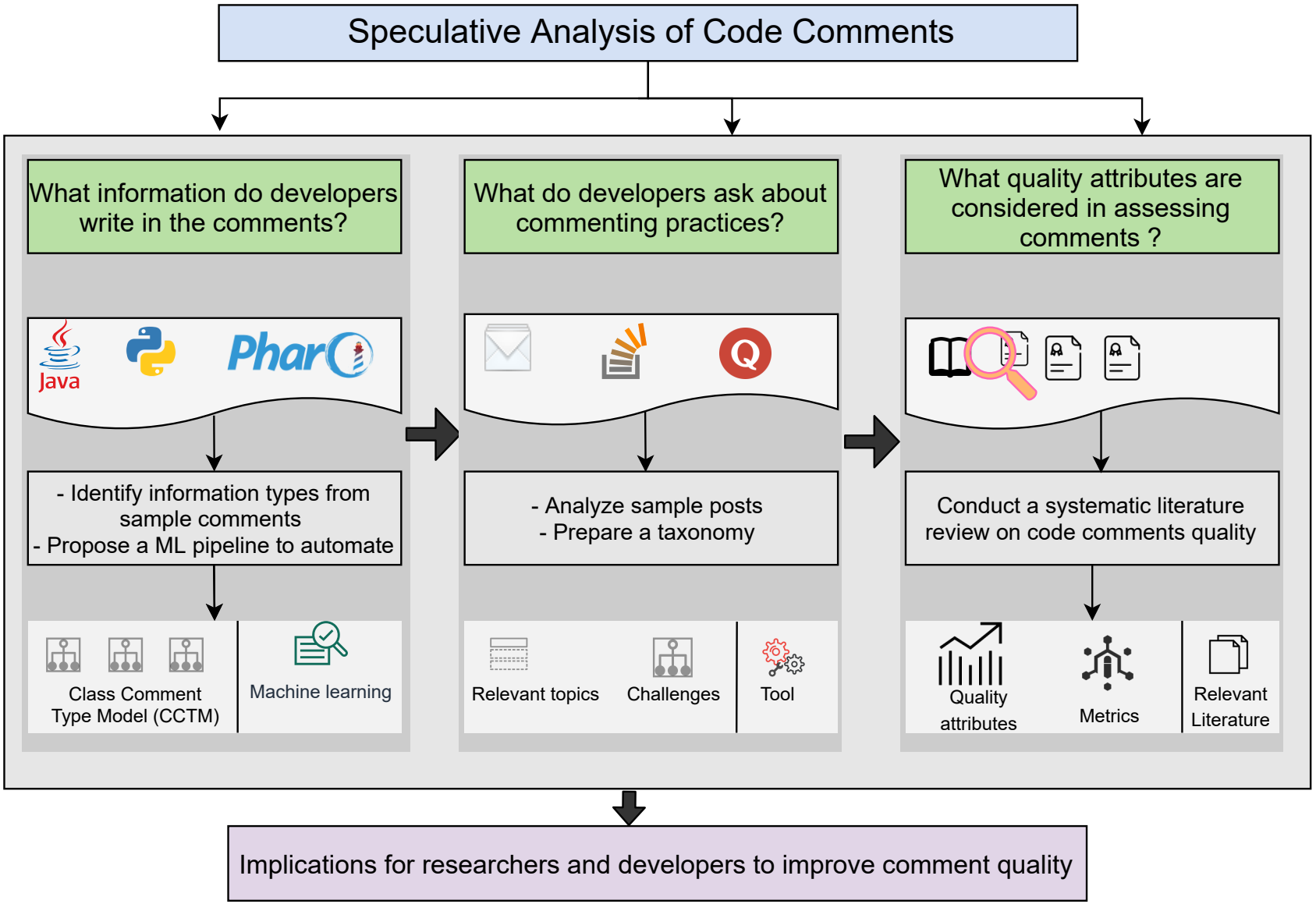
Results are submitted to a software engineering venue

Do Developers Follow Comment Conventions?

An investigation of comment adherence to comment conventions

In preparation

In the stage of submission


Speculative Analysis of Code Comments
What information do developers write in the comments?
Java
Pharo
- Identify information types from sample comments
- Propose a ML pipeline to automate
Class Comment Type Model (CCTM)
Machine learning
What do developers ask about commenting practices?
Q
- Analyze sample posts
- Prepare a taxonomy
Relevant topics
Challenges
Tool
What quality attributes are considered in assessing comments ?
Conduct a systematic literature review on code comments quality
Quality attributes
Metrics
Relevant Literature
Implications for researchers and developers to improve comment quality

The ultimate goal of automatically assessing comments is still far away…

Future work

Which quality attributes do developers find important?

Which information types do developers find important?

How do various information types support developers?

An IDE plugin to support automatic assessment of comments

Takeaway

Developer seek and follow comment conventions.

Having support to assess various aspects of comments can help maintaining high-quality comments

https://twitter.com/poojaruhal

http://scg.unibe.ch/staff/Pooja-Rani