# Code Comments and Software repository mining research

Kickoff meeting

Kam Srisopha
Thammanoon Kawinfruangfukul

# Agenda

- Ice-breaking
- Motivation
- Research Proposals
- Weekly Meeting
- Expectation

Let’s get to know each other!

# Developers use code comments for multiple reasons


```
                }// end of for loop
        }// end of while loop
}//end of function call
```

```
//don't change anything after this line
```

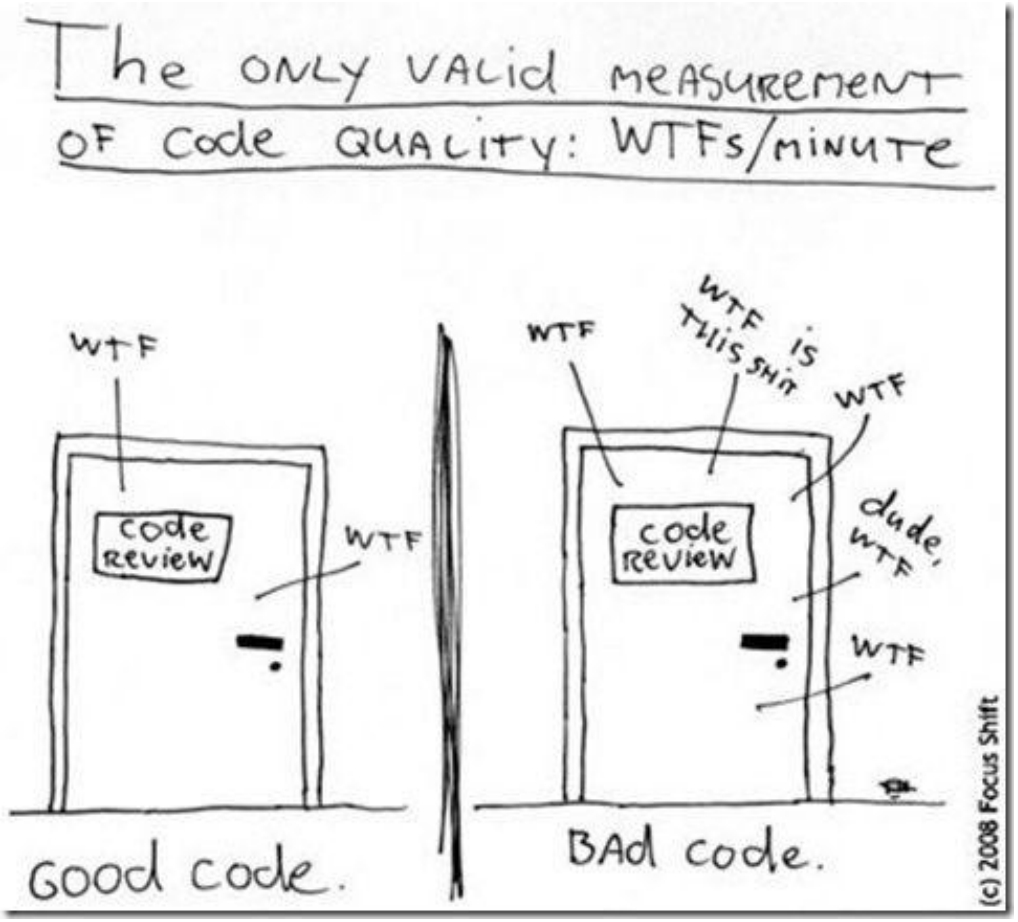
```
//TODO: this is a hack, should fix in the next iteration
```

```
//Bug 323424 <some link>
```

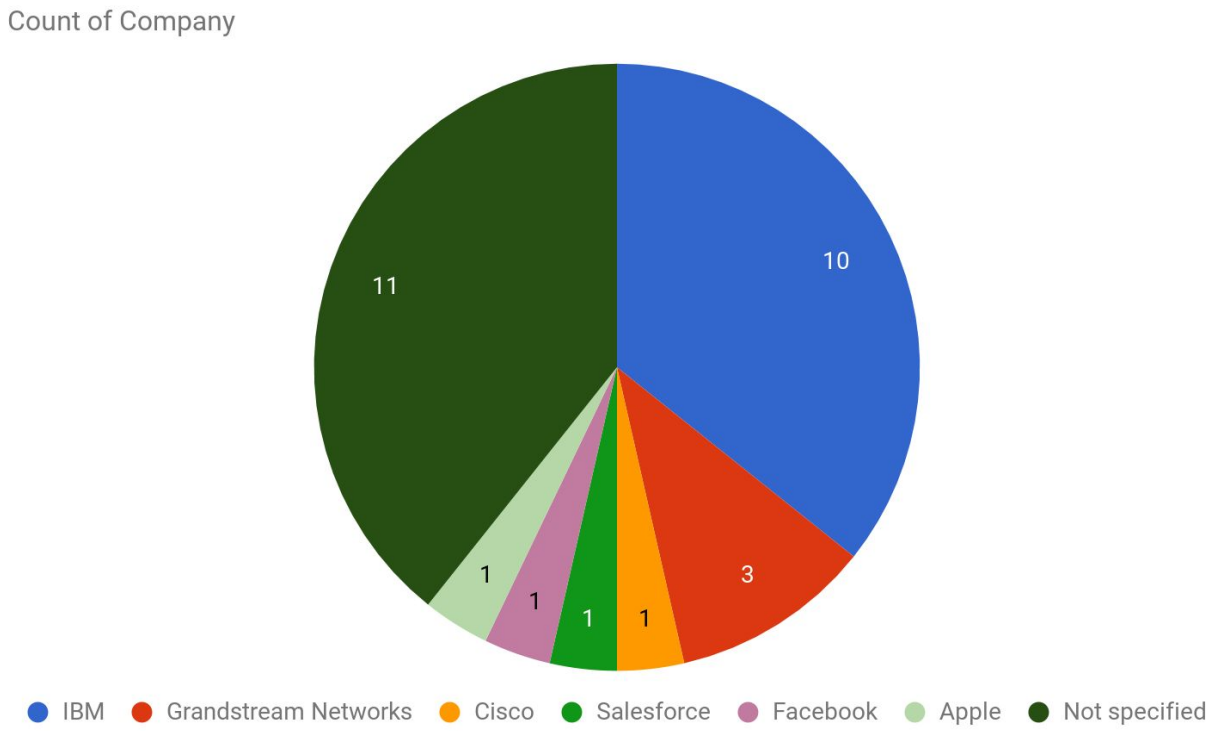
```
//System.out.println("This is for debugging");
//System.out.println("Here!");
```

**And many many more….**

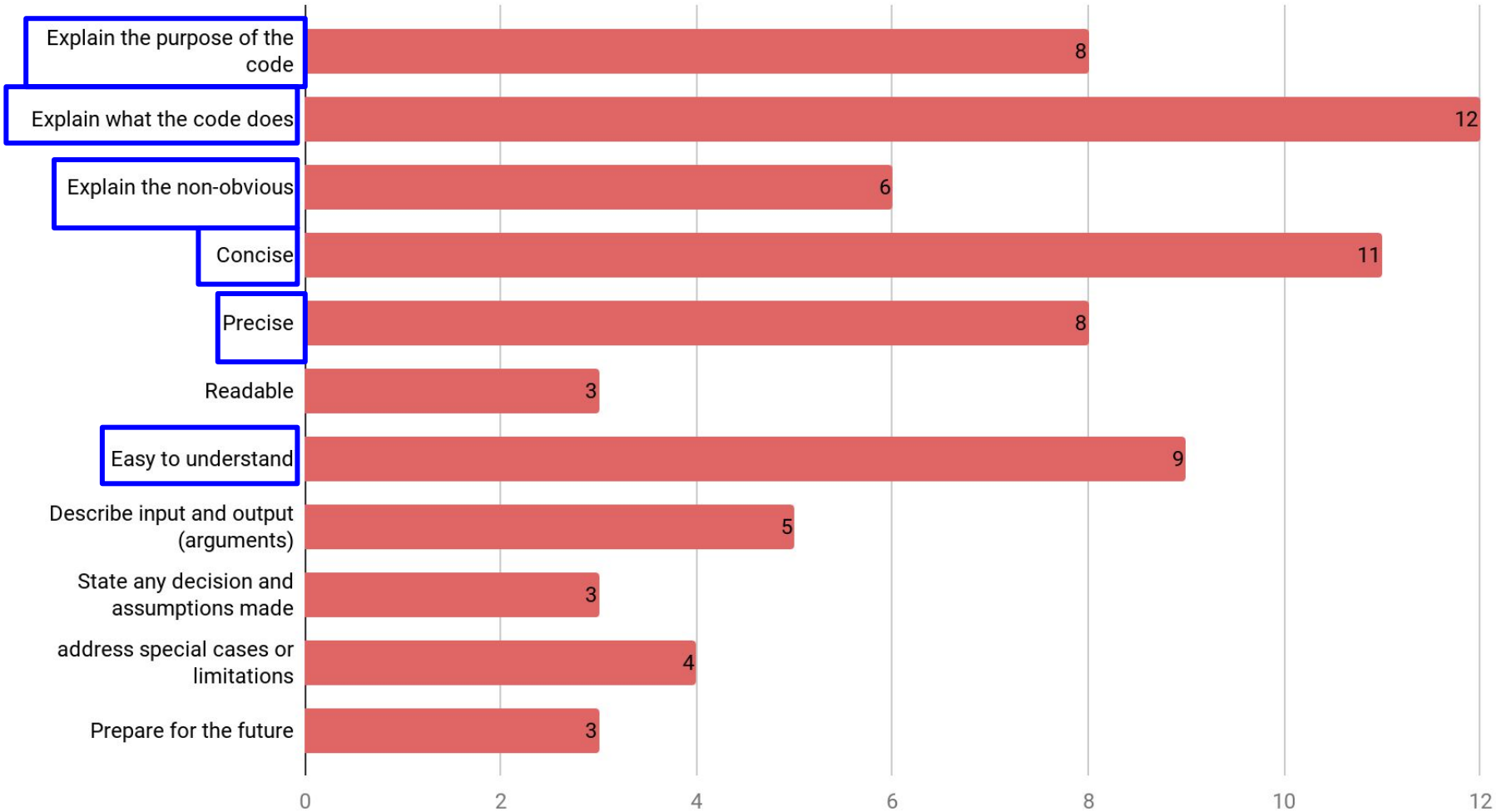
**Good/Valuable Code Comment affects software quality, especially program clarity, readability, maintainability, etc.**



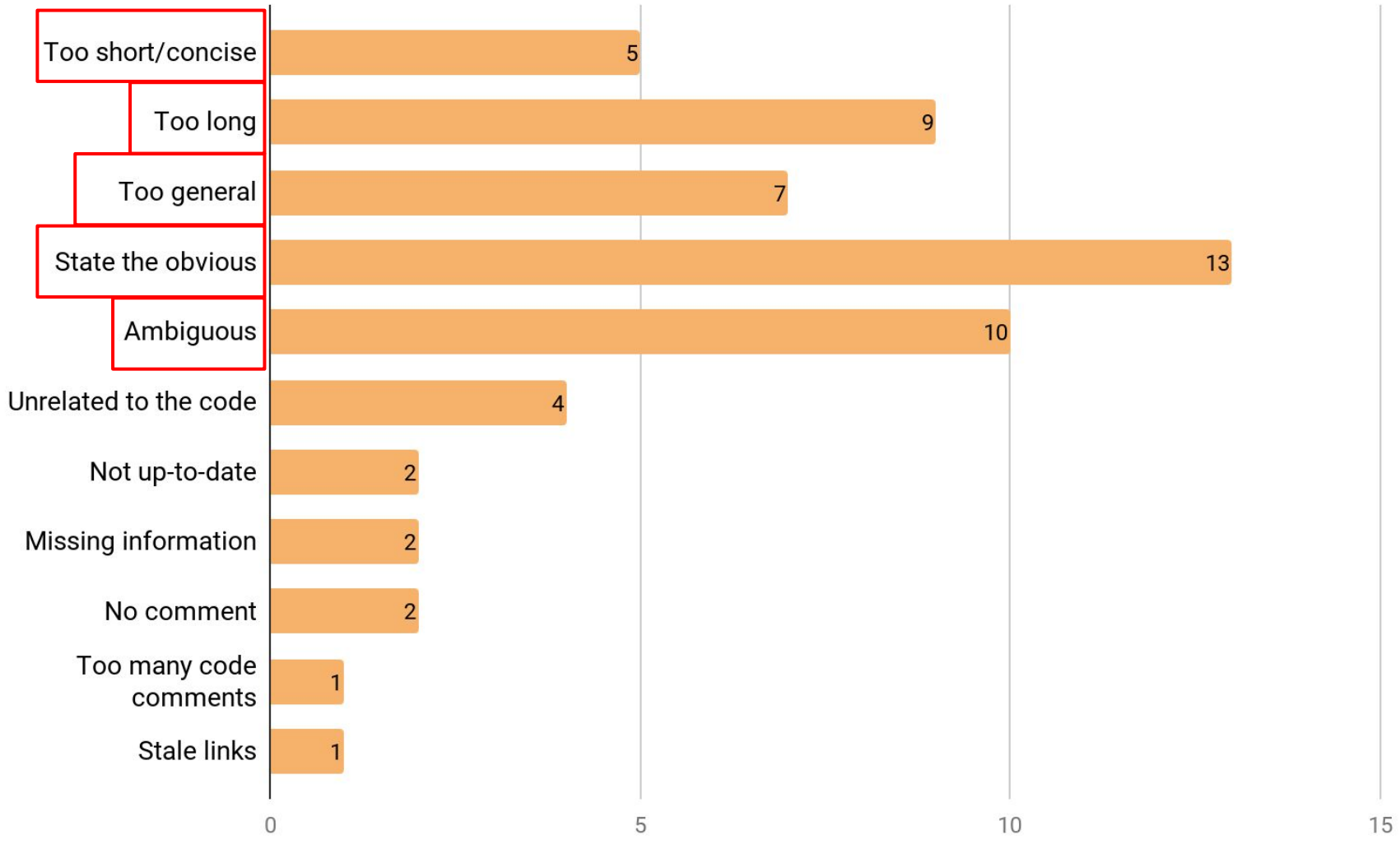
# **Exploratory study:** Survey 28 developers in the industry on code comments



# Q1: What are the characteristics of useful code comments



# Q2: What characteristics are indicative of a low-quality code comment



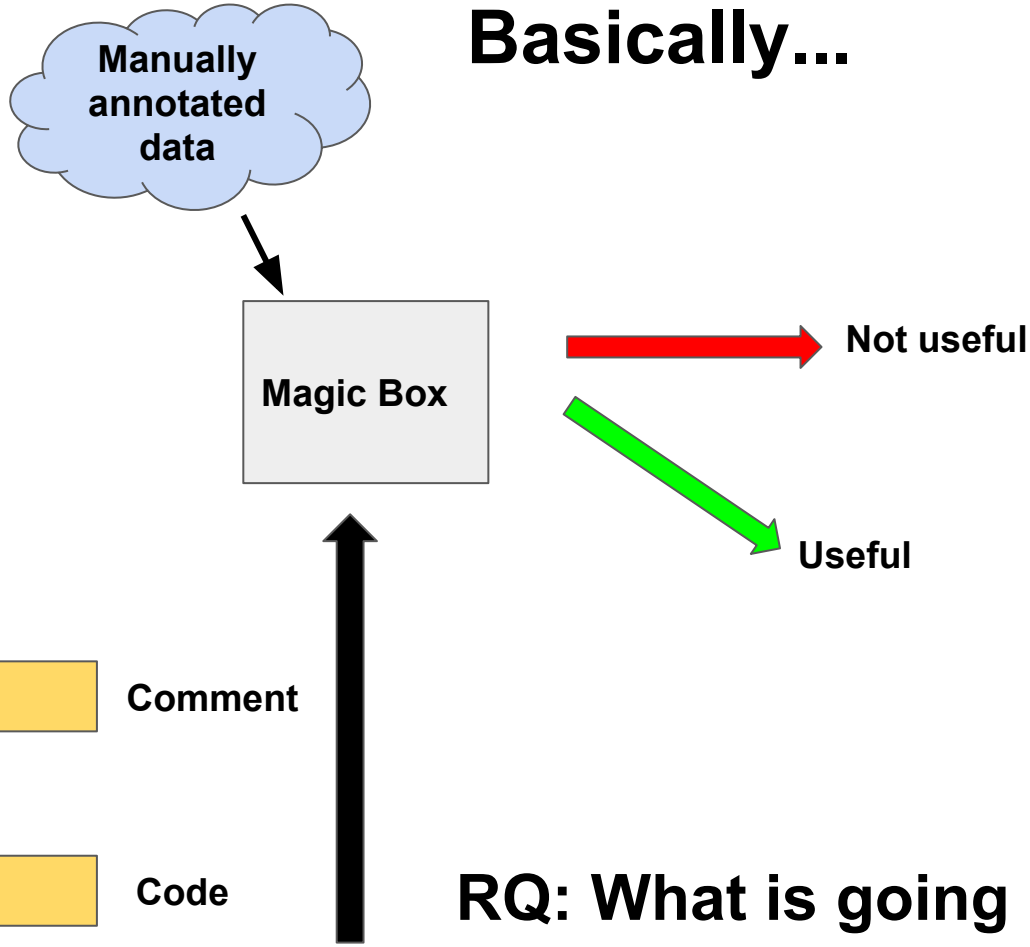
# So what are the good code comments?

- Should explain the non-obvious
  - Should not be redundant
- Should not be too long or too short
- Should not be ambiguous
  - RQ: What makes code comments ambiguous?
- Should be precise and up-to-date
  - RQ: How to check if the code is precise and up-to-date?
- Should explain what the code does

```
/**
 * Unload a decoder module
 */
static void UnloadDecoder( decoder_t *p_dec )
{
    if( p_dec->p_module )
    {
```

**“UnCamelCasing”**

**Project #1:** Automatically assess/predict usefulness of code comments


**Basically...**

**RQ: What is going on inside that magic box?**

# What can go into the magic box?

**Buzzwords**

- Fuzzy rules
- Wordnet/ Word2Vec
- Feature Extractions ( based on the survey responses + literature reviews)
- Different supervised machine learning techniques
- Unsupervised machine learning techniques??

**Challenging questions:**

- How do we check if the code and its comment are coherent?
  - Readily available coherence data from prior study: http://www2.unibas.it/gscanniello/coherence/
  - Information retrieval
    "Recovering Traceability Links between Code and Documentation" paper
- How can we understand what the code does?

# Ground Truth (manually annotate code comments)

**We already created a tool to help with that**

CSSE Annotator Home test@mail.com

**File:** java_src_1.java **This file contains** 14 comments **You have done** 1 / 2 files Request New File

Code </>

```
1  // my first java program
2  public class java_src_1 {
3
4  }
5  //     public static void main(String[] args) {
6  //         System.out.println("Hello, World!");
7  //     }
8  /* this program prints:
9   * Hello, World!
10  * to the console */
11 // }
12
13 /*Test match in one line*/
14
15 string = "//hi there";
16
17 // hey, /this doesn't fail!/ :D
18
19 // http://lea.verou.me
20 /* http://lea.verou.me */
21
22 // http://lea.verou.me
23 var comment = "/\/\*[\w\W]*?\*\//g";
```

Comment #1

// my first java program

Select Category:

Valuable Somewhat valuable Not valuable

Coherence: Yes No

Link source to comment: Ex. 2-4 or 4 for comment #1

Explanation (optional): Free text input for comment #1

Group with comment (optional): None selected

« Prev Next » Submit Response

# Ground Truth

**<u>2 ways to do it correctly (what researchers use):</u>**

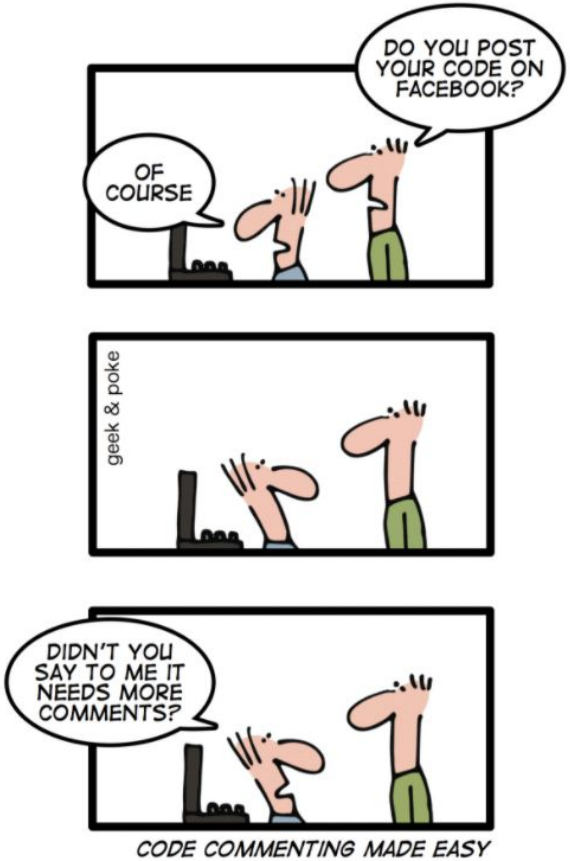
1). Everyone classifies everything in the dataset and then disputes any discrepancies.

2). Do a dry run with 100 randomly sample data (everyone does it) then use

**Cohen's Kappa** coefficient to assess the level of agreement
(if we get ~1) then we can say we are in complete agreement.

Then each will do annotate separately and combine everything as a big dataset at the end.

# **Project #2:** Automatically generate code comments based on code




**RQ:** How do we actually know what the code does? Maybe based on requirements, api documentation, AST tree, IR?

**Most prominent plug-in: *Ghostdoc* for C#**
**But….**

**Kind of redundant**

```
public class Person
{
    /// <summary>
    /// Initializes a new instance of the <see cref="Person"/> class.
    /// </summary>
    public Person() {}
}
```

**What does that even mean?**

```
/// <summary>
/// Riches the text selection changed.
/// </summary>
/// <param name="richTextBox">The rich text box.</param>
private void RichTextSelection_Changed(System.Windows.Controls.RichTextBox richTextBox)
```

**Q:** Can we use LSTM (recurrent neural network) to help us somehow?

**See:** Commit message auto generator from neural machine translation (neural network)
https://arxiv.org/abs/1708.09492
https://arxiv.org/abs/1703.09603

# **Project #3:** Classifying different types of self-admitted technical debt from either commit message or code comments

**See: " Towards an Ontology of Terms on Technical Debt" paper for different types of technical debt**

1. **Design Debt**
2. **Test Debt**
3. **Documentation Debt**
4. **Code Debt**
5. **Defect Debt**
6. **Requirement Debt**

*"TODO: - This method is too complex, lets break it up"* - [from ArgoUml]

*"/* TODO: really should be a separate class */"* - [from ArgoUml]

*"**FIXME** This function needs documentation"* - [from Columba]

*"// TODO Document the reason for this"* - [from Apache Jmeter]

*"/TODO no methods yet for getClassname"* - [from Apache Ant]

**Opportunity:**

- One paper does it for 2 categories ( Design debt and Requirement Debt) using only NLP -2017
- One paper uses 63 pattern matching (63 words likely to be in SATD comments) - 2016
- https://www.youtube.com/watch?v=Baf18V6sN8E
- Further classification for code smells

**Anything in particular (related to the topic) that you want to do?**

# Weekly Meeting + Individual Meeting

# Expectation


- These projects are research projects, not software development projects. That means that everything might not work 100%, which is okay as long as you show that you try something!

- I want you to expose yourself with the newest CS techniques e.g., machine learning techniques

- “Be like a bulldog” - never back down, keep attacking the problems and have fun :)