Maintainability Index Variation Among PHP, Java, and Python Open Source Projects

Celia Chen[1], Lin Shi[2], Kamonphop Srisopha[1]

[1] Computer Science Department, USC

[2] Laboratory for Internet Software Technologies, ISCAS

Agenda

Motivation

Maintainability Index

Research Question

Data Collection

Results

Conclusion

Future Work

Motivation

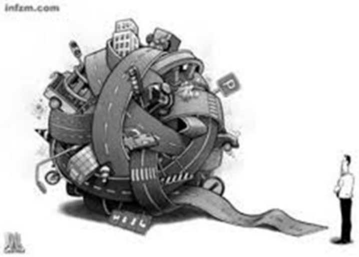
- Open Source Projects
  - Global Distributed Collaboration
  - Voluntarily

Suède
Norvège
Canada
Mongolie
Ouzbékistan
Libye
Angola
Namibie
Bolivie
Paraguay
Australie

- Low maintainability

Difficult to modify

Increase the participation cost



Difficult to find solutions for bugs

Increase the maintainability effort

Motivation

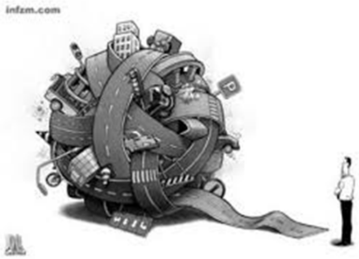
- Open Source Projects
  - Global Distributed Collaboration
  - Voluntarily

- Low maintainability

A successful OSS project requires to be highly maintainable

Difficult to modify

Increase the participation cost



Difficult to find solutions for bugs

Increase the maintainability effort

Why Programming Languages？

"C makes it easy to shoot yourself in the foot; C++ makes it harder, but when you do it blows your whole leg off." — Bjarne Stroustrup

- Impact of the language choice is significant
  - "like choosing a wife" — *Barry W. Boehm*
  - Impact on design, development, later maintenance phases

Our goal: investigate the impact of programming language on maintainability

Maintainability

- “The **ease** in which a system can be modified or extended”
- **Maintainability Index (MI)**
  - An index that represents the ease of maintaining the code
  - Widely used in the industry

Microsoft® Visual Studio®

JHAWK6

Verifysoft TECHNOLOGY

PHPMETRICS

Static analysis tool for PHP

Maintainability Index

$$MIwoc_{(sourcefile)} = 171 - 5.2 * \ln HV - 0.23 * CC - 16.2 * \ln LLOC$$

$$MIwc_{(sourcefile)} = 50 * sin\sqrt{2.46 * CM}$$

$$MI_{(sourcefile)} = MIwoc_{(sourcefile)} + MIwc_{(sourcefile)}$$

$$MI = \frac{\sum MI_{(sourcefile)}}{Number\ of\ Source\ files}$$

Halstead Volume (HV)
Count of lines (LLOC)

Cyclomatic complexity (CC)
Percent of lines of comments (CM)

MI is developed by the University of Idaho in 1991 by Oman and Hagemeister

Halstead Volume

According to Halstead, a computer program is an implementation of an algorithm considered to be a collection of tokens which can be classified as either operators or operands.

Operators include:

Reserved words (while, if, do, class, etc)
Qualifier (const, static)
expressions and arithmetic operators (+, >,=)
etc.

Operand includes:

numeric constant
literal
identifiers
etc.

n1 = number of distinct operator
n2 = number of distinct operands
N1 = Total number of occurrences of operators
N2 = Total number of occurrences of operands

Program Length: N = N1 + N2
Vocabulary Size: n = n1 + n2

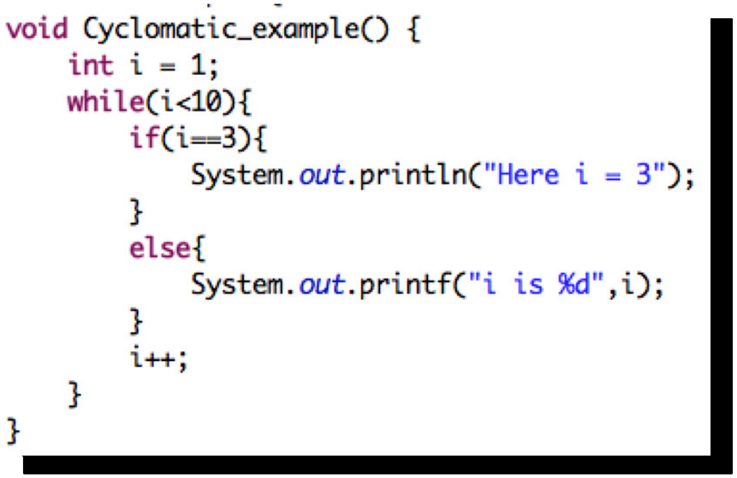
Program Volume = N * $\log_2(n)$

McCabe’s Cyclomatic Complexity

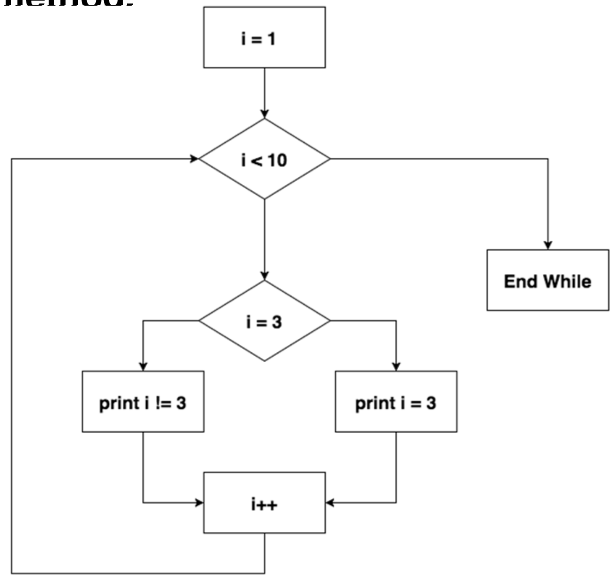
Cyclomatic Complexity aims to capture the complexity of a code function/method in a single number. The metric develops a Control Flow graph that measures the number of linearly independent paths through a program module*

$$CC = E - N + 2 \times P$$

E = number of edges
N = number of nodes
P = number of module/ connected function/method.



```
void Cyclomatic_example() {
    int i = 1;
    while(i<10){
        if(i==3){
            System.out.println("Here i = 3");
        }
        else{
            System.out.printf("i is %d",i);
        }
        i++;
    }
}
```



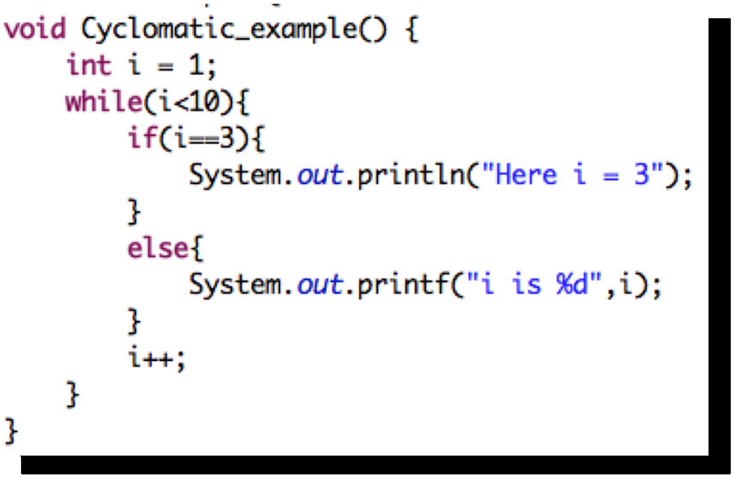
*http://www.tutorialspoint.com/software_testing_dictionary/cyclomatic_complexity.htm

McCabe’s Cyclomatic Complexity

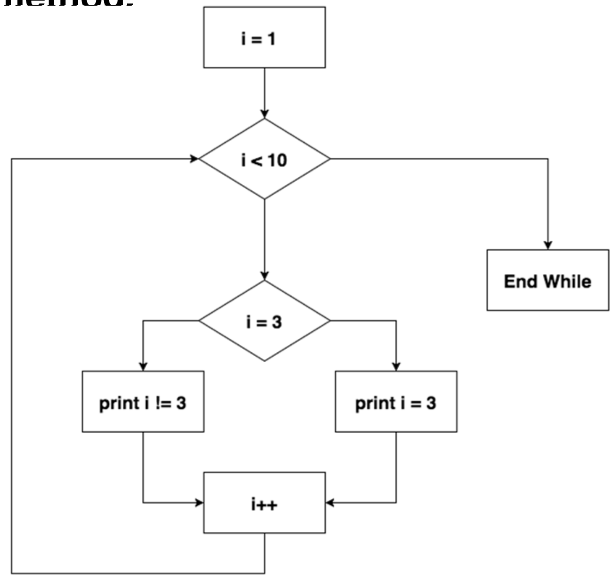
Cyclomatic Complexity aims to capture the complexity of a code function/method in a single number. The metric develops a Control Flow graph that measures the number of linearly independent paths through a program module*

$$CC = E - N + 2 \times P$$

E = number of edges
N = number of nodes
P = number of module/ connected function/method.



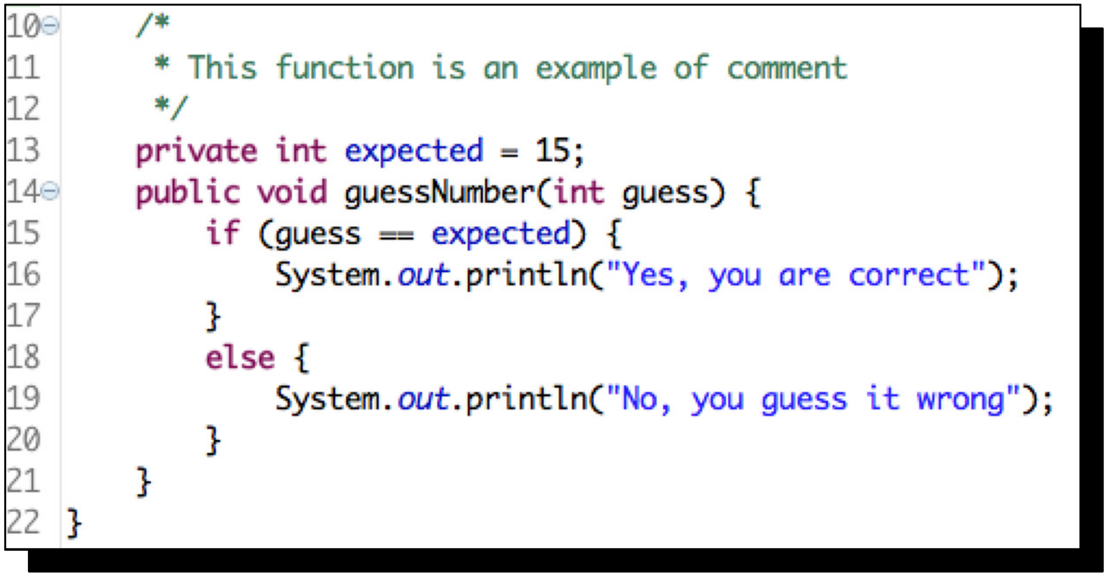
```
void Cyclomatic_example() {
    int i = 1;
    while(i<10){
        if(i==3){
            System.out.println("Here i = 3");
        }
        else{
            System.out.printf("i is %d",i);
        }
        i++;
    }
}
```



$$CC = 8 - 7 + (2 * 1) = 3$$

*http://www.tutorialspoint.com/software_testing_dictionary/cyclomatic_complexity.htm

Logical Line of Code



```
    /*
     * This function is an example of comment
     */
    private int expected = 15;
    public void guessNumber(int guess) {
        if (guess == expected) {
            System.out.println("Yes, you are correct");
        }
        else {
            System.out.println("No, you guess it wrong");
        }
    }
}
```

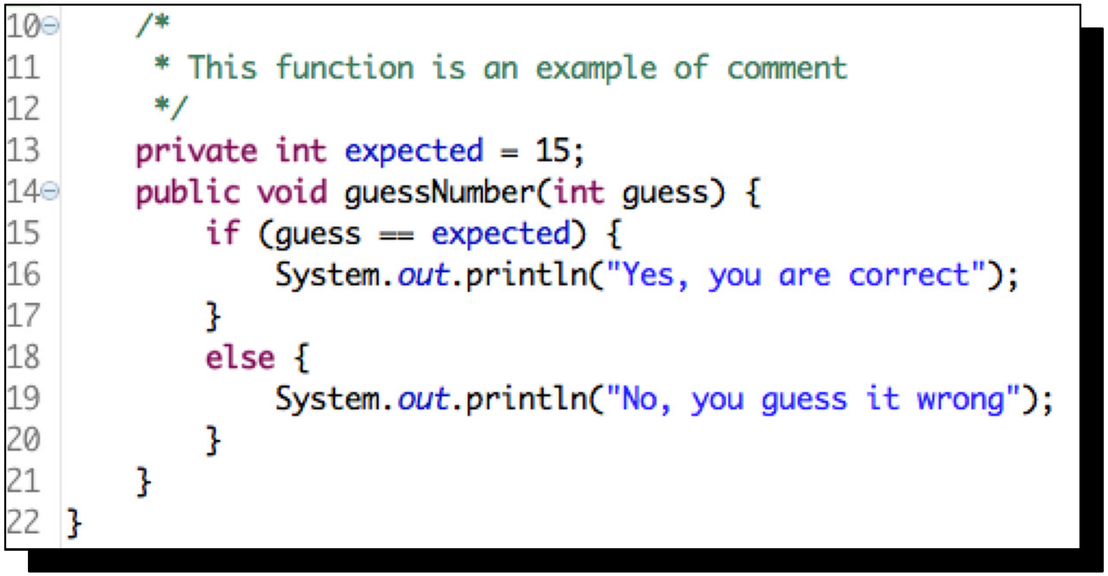
Logical Line of Code attempts to measure the number of executable expression/ statements

Physical Line of Code

Logical Line of Code

Comment

Logical Line of Code



```
    /*
     * This function is an example of comment
     */
    private int expected = 15;
    public void guessNumber(int guess) {
        if (guess == expected) {
            System.out.println("Yes, you are correct");
        }
        else {
            System.out.println("No, you guess it wrong");
        }
    }
}
```

Logical Line of Code attempts to measure the number of executable expression/ statements

Physical Line of Code	**13**
Logical Line of Code	**6**
Comment	**3**

First Research Question

How does MI vary among Java, PHP, and Python open source projects?

Language Hypothesis

For PHP, Java and Python OSS projects, MI varies significantly.

Null Hypothesis

MI does not vary significantly across PHP, Java and Python OSS projects.

Second Research Question

Does MI vary among various domains for these open source projects?

If yes, does language choice affect MI within each domain?

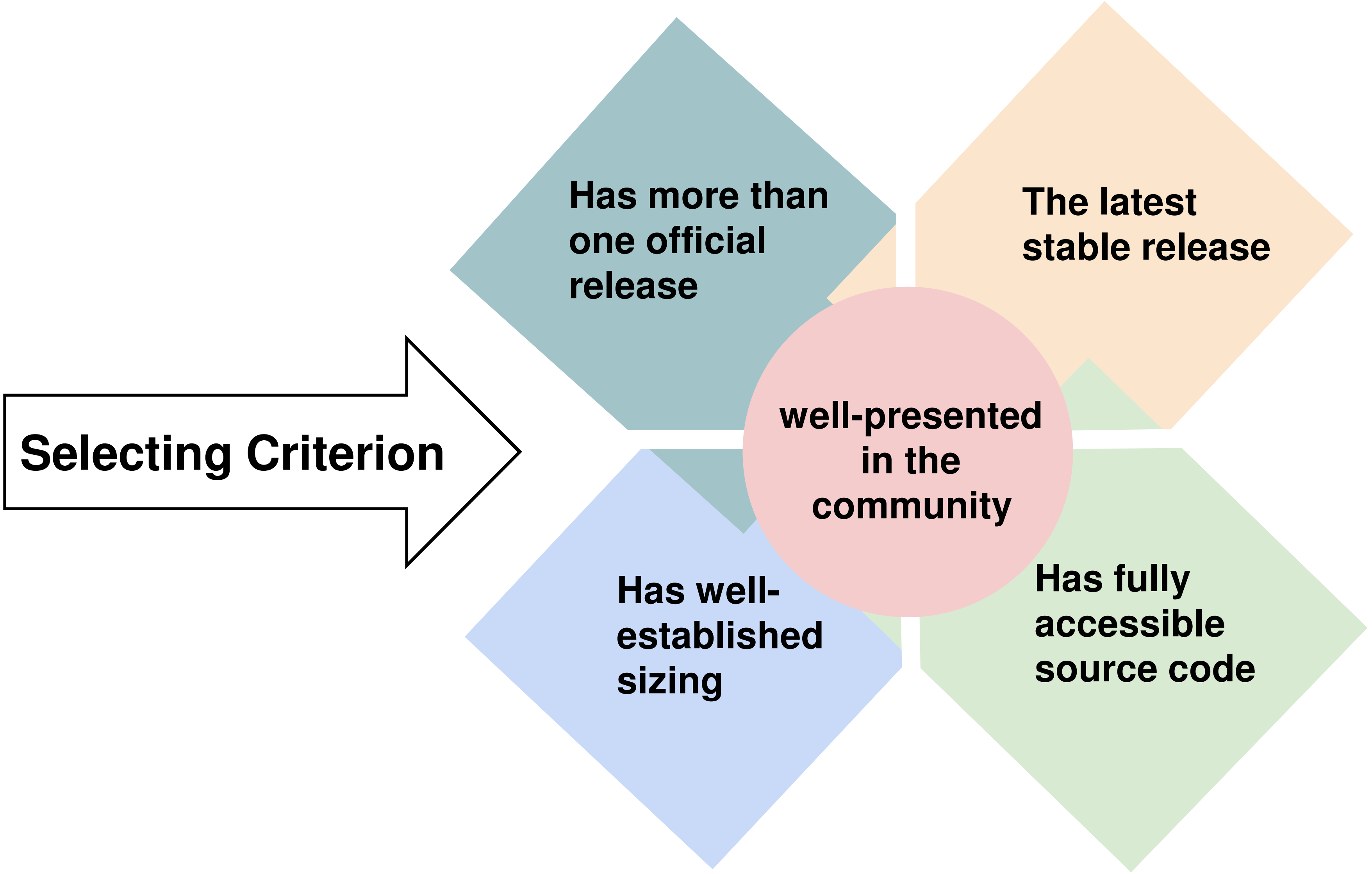
Domain Hypothesis

For different software development domains, MI of PHP, Java and Python OSS projects varies significantly

Null Hypothesis

MI does not vary significantly across different software development domains

Data Collection



Selecting Criterion

Has more than one official release

The latest stable release

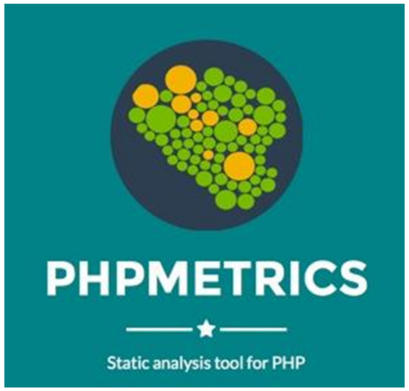
well-presented in the community

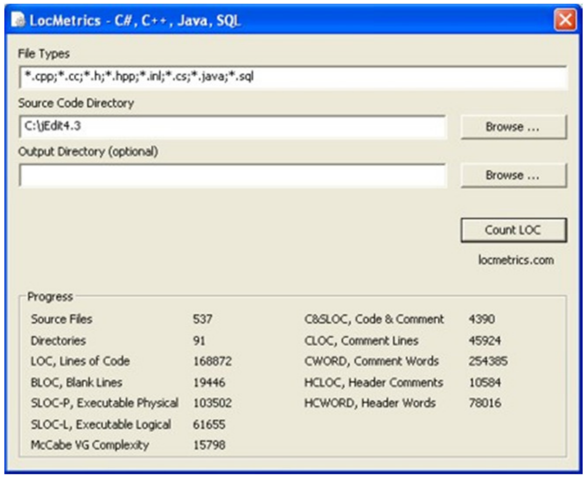
Has well-established sizing

Has fully accessible source code

Characteristics of project data sources

Language	Average LLOC	Metrics Collection Tools
PHP	18643	Phpmetrics
Java	33871	CodePro, LocMetrics
Python	6644	Radon





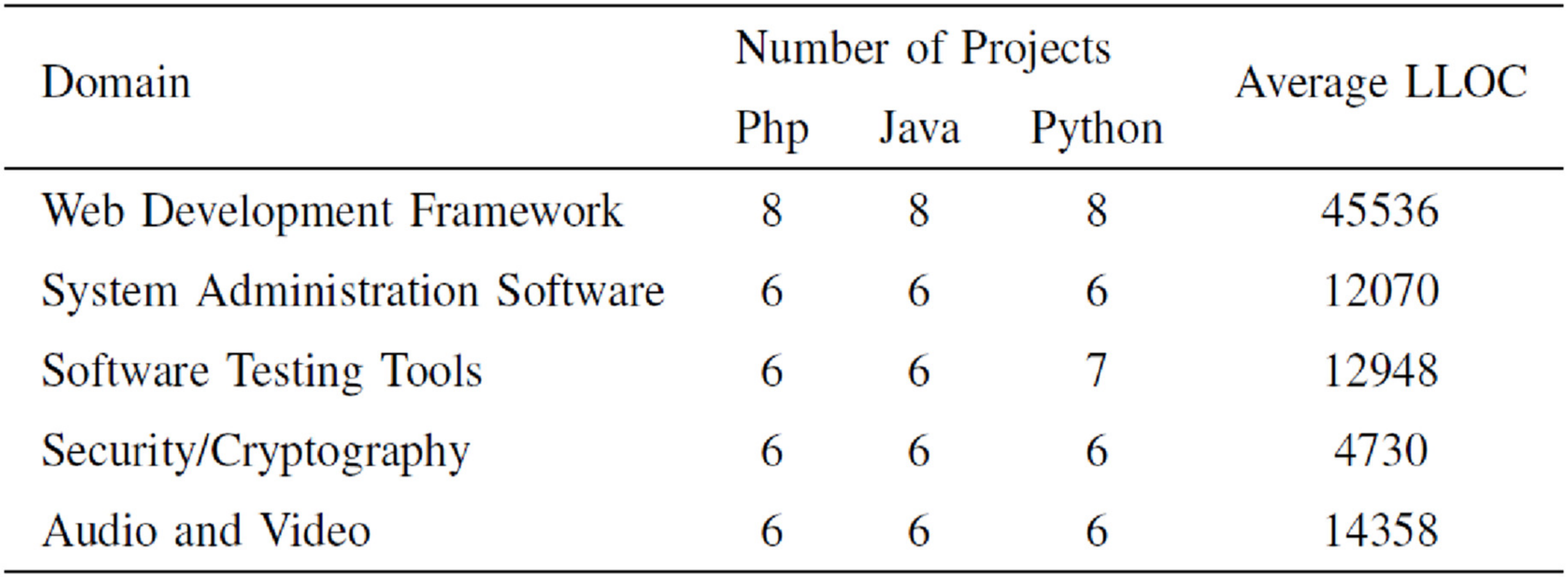
File Types: *.cpp;*.cc;*.h;*.hpp;*.inl;*.cs;*.java;*.sql

Source Code Directory: C:\jEdit4.3

Output Directory (optional):

```
Progress
Source Files                  537      C&SLOC, Code & Comment    4390
Directories                   91       CLOC, Comment Lines       45924
LOC, Lines of Code            168872   CWORD, Comment Words      254385
BLOC, Blank Lines             19446    HCLOC, Header Comments    10584
SLOC-P, Executable Physical   103502   HCWORD, Header Words      78016
SLOC-L, Executable Logical    61655
McCabe VG Complexity          15798
```

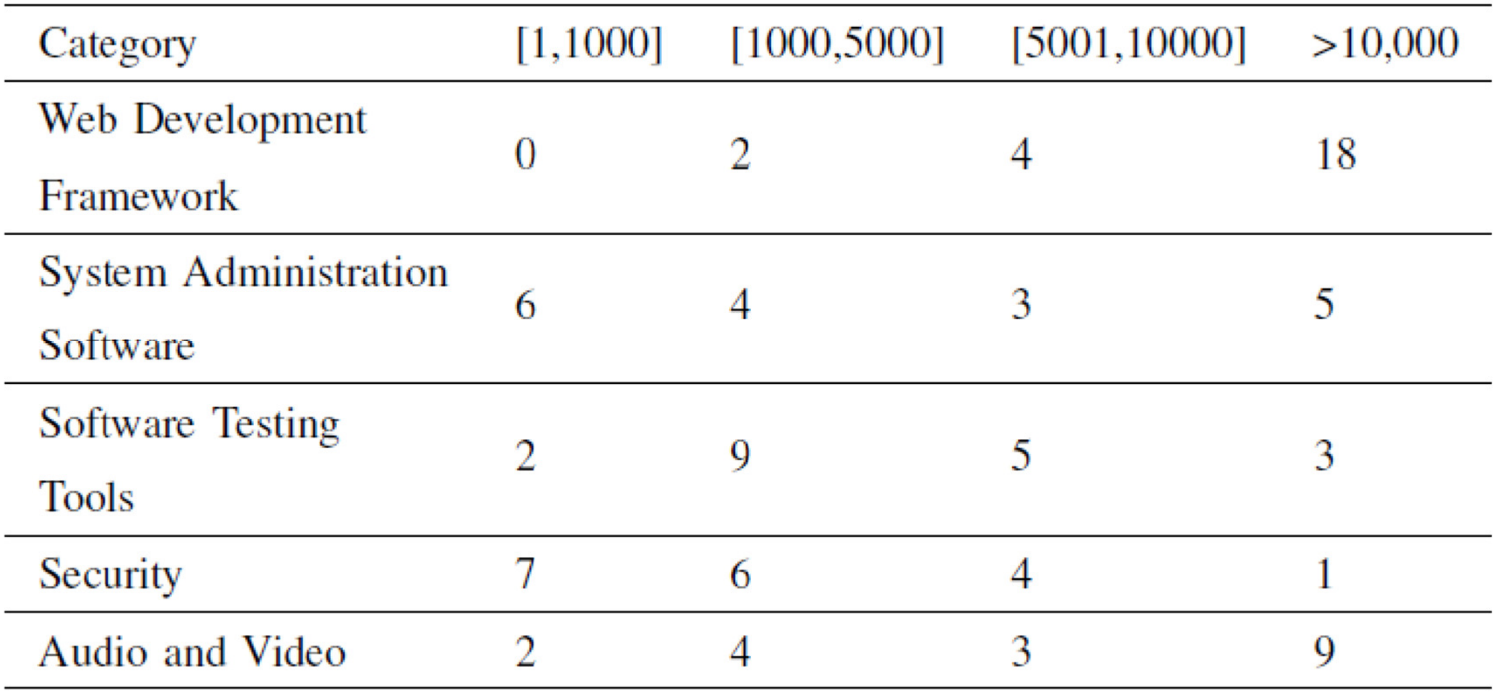
Characteristics of project domains



Domain	Number of Projects			Average LLOC
	Php	Java	Python	
Web Development Framework	8	8	8	45536
System Administration Software	6	6	6	12070
Software Testing Tools	6	6	7	12948
Security/Cryptography	6	6	6	4730
Audio and Video	6	6	6	14358

*** Excluding test, doc, example, tutorial folders**

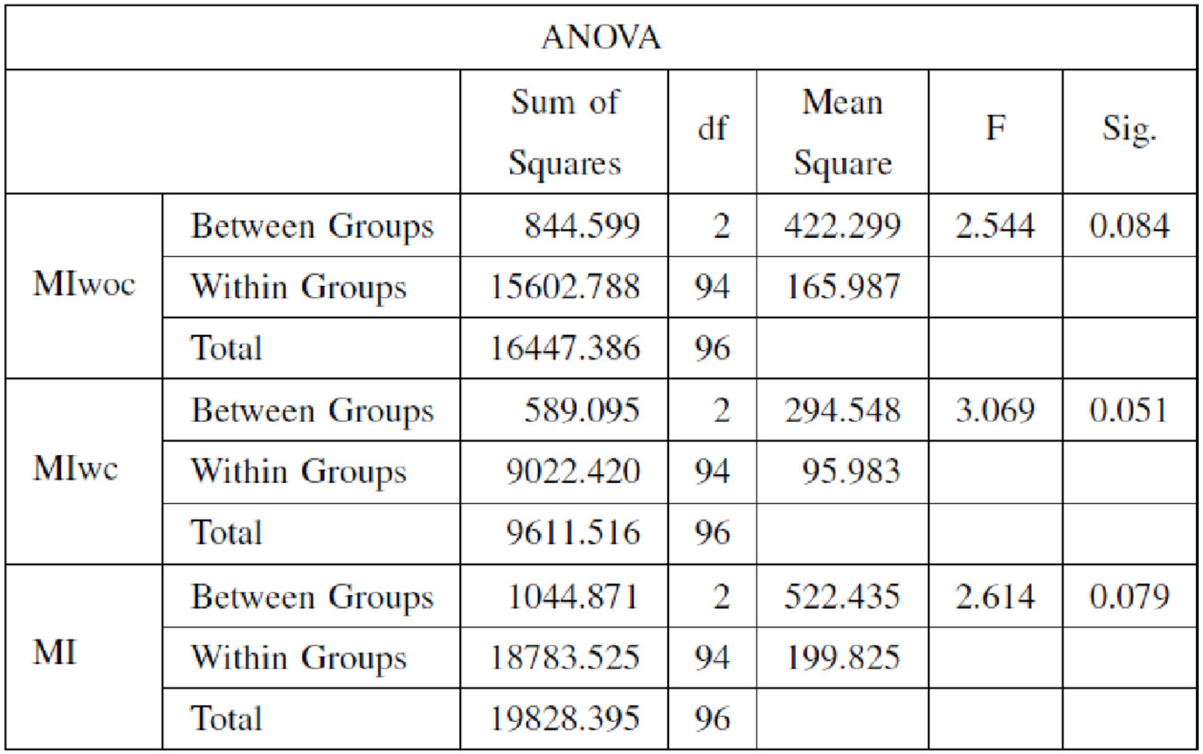
Classification on number of projects by LLOC in each domain



Category	[1,1000]	[1000,5000]	[5001,10000]	>10,000
Web Development Framework	0	2	4	18
System Administration Software	6	4	3	5
Software Testing Tools	2	9	5	3
Security	7	6	4	1
Audio and Video	2	4	3	9

Results – RQ1

One-way ANOVA Results for language analysis

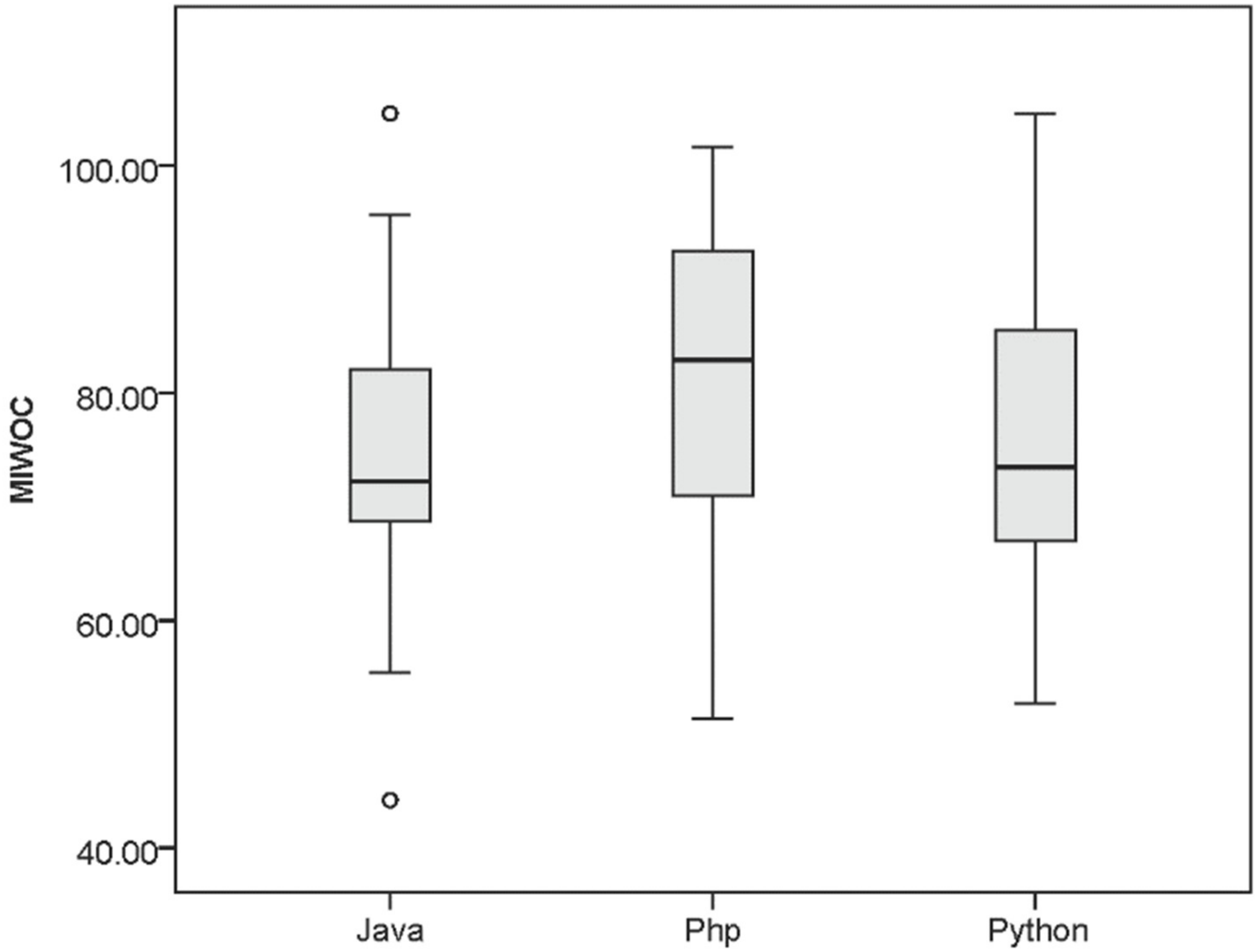


ANOVA

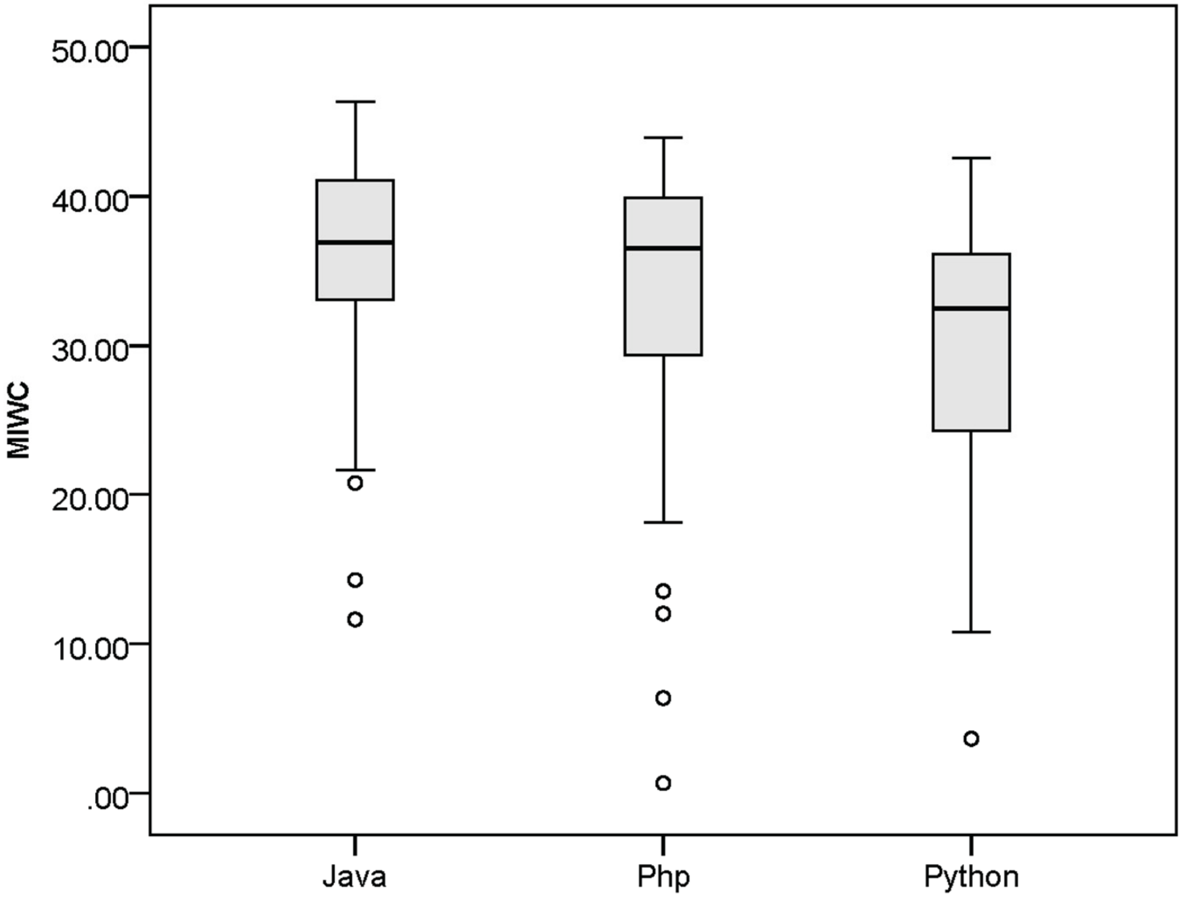
		Sum of Squares	df	Mean Square	F	Sig.
MIwoc	Between Groups	844.599	2	422.299	2.544	0.084
	Within Groups	15602.788	94	165.987		
	Total	16447.386	96			
MIwc	Between Groups	589.095	2	294.548	3.069	0.051
	Within Groups	9022.420	94	95.983		
	Total	9611.516	96			
MI	Between Groups	1044.871	2	522.435	2.614	0.079
	Within Groups	18783.525	94	199.825		
	Total	19828.395	96			

- P-Value <0.1 (Strongly suggestive)
  - MI differs across the three languages at 90% confidence level
  - Reject Null Hypothesis

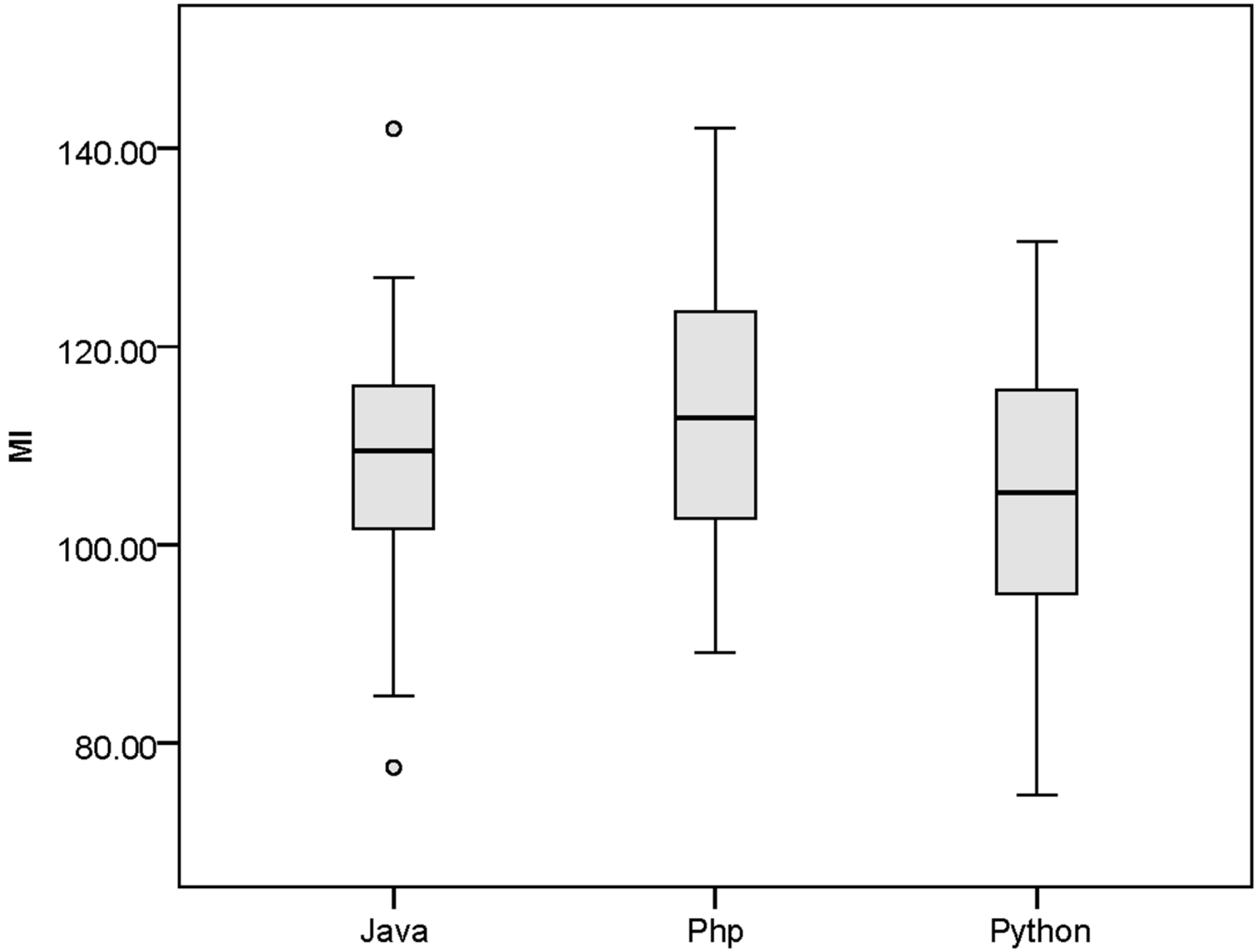
Maintainability Index without comment (MIWOC)



Maintainability Index with comment (MIWC)

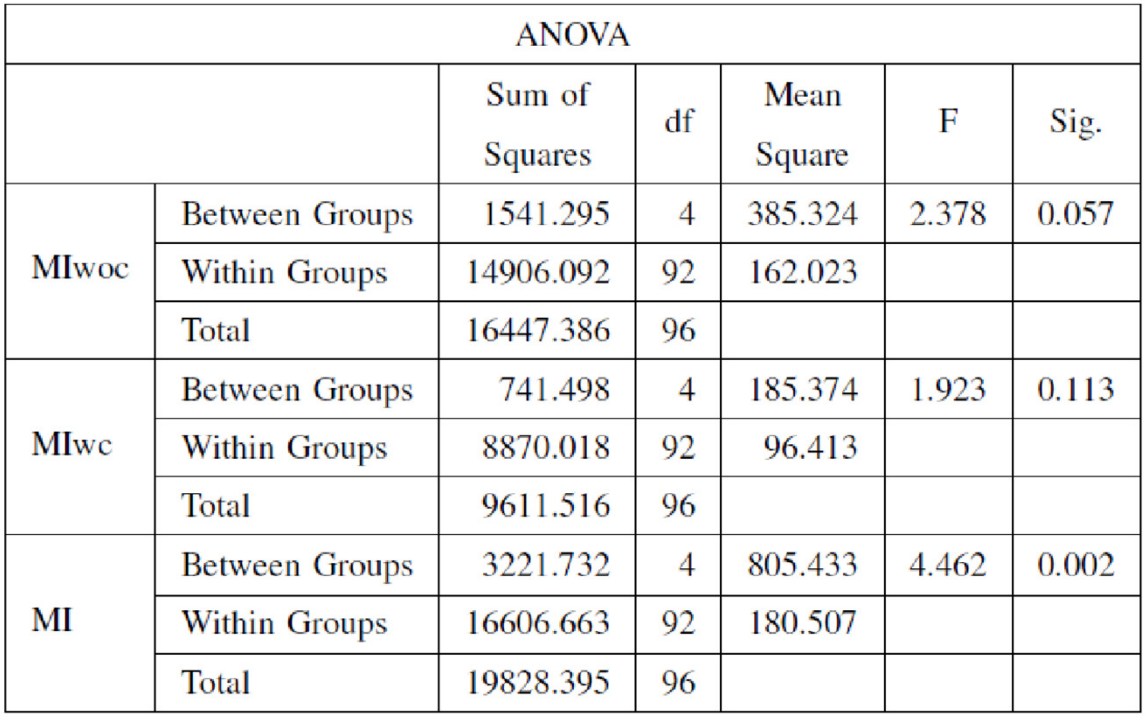


Maintainability Index = MIWOC + MIWC



Results – RQ2

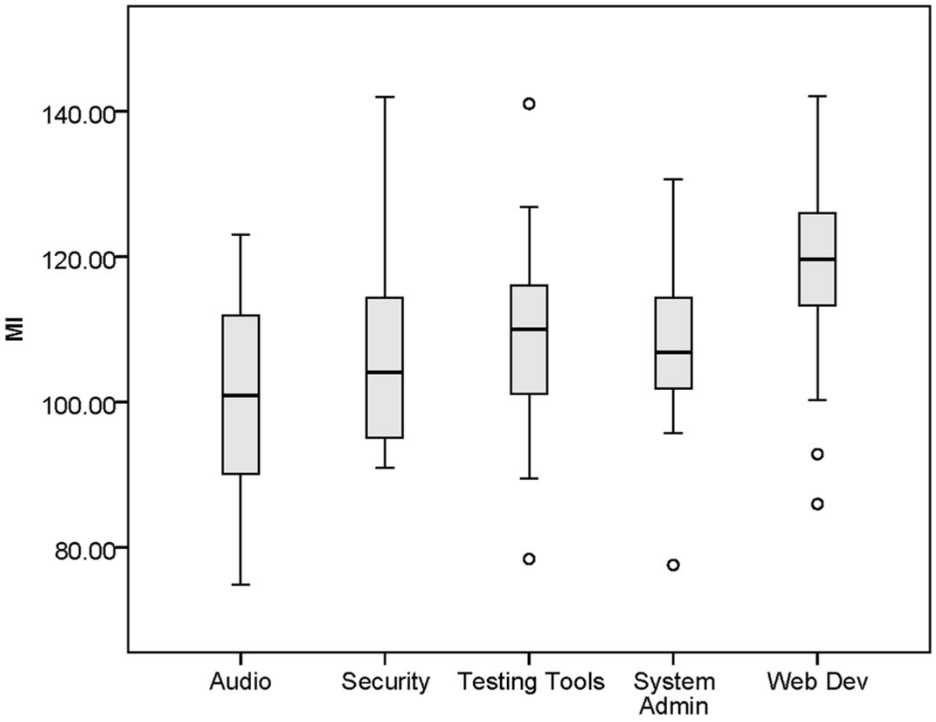
One-way ANOVA for domains



ANOVA						
		Sum of Squares	df	Mean Square	F	Sig.
MIwoc	Between Groups	1541.295	4	385.324	2.378	0.057
	Within Groups	14906.092	92	162.023		
	Total	16447.386	96			
MIwc	Between Groups	741.498	4	185.374	1.923	0.113
	Within Groups	8870.018	92	96.413		
	Total	9611.516	96			
MI	Between Groups	3221.732	4	805.433	4.462	0.002
	Within Groups	16606.663	92	180.507		
	Total	19828.395	96			

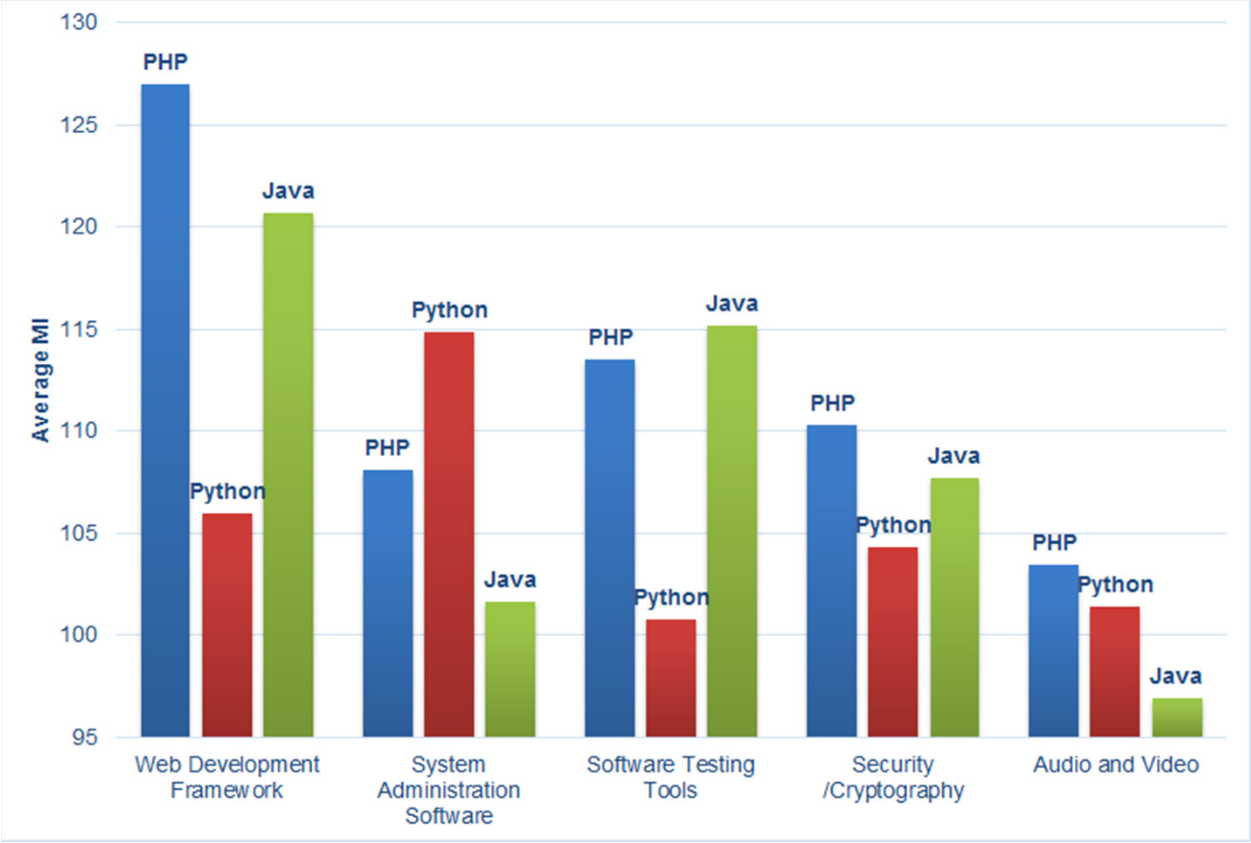
- P-Value <0.05 (Definitive)
  - MI differs across the five domains at 95% confidence level
  - Reject Null Hypothesis

MI Variation among domains



- Web Development Framework has shown the highest medians and the highest maximum value.
- Audio and Video has both the lowest maximum value and the lowest median value

Average MI for each Language



- **PHP** may be a good option for projects that desires higher maintainability within Web Development Framework, Security/Cryptography and Audio and Video domain,
- **Python** may be a good option for System Administrative Software
- **Java** for Software Testing Tools**.**

Maintainability Index — To be Improved

- Maintainability Index only consider **Code Quality** (Halstead Volume, Cyclomatic complexity), **Size** (Count of lines), and **Comments Ratio** as indicators.

- To comprehensively and accurately indicate the ease to maintain for OSS, **there are more aspects need to be considered**:

For example:

Code Structure: Cohesion & Coupling
Application Clarity
Documentation Quality
Community Support

Conclusion

- Based on a dataset of 97 open source projects,
  - Employed one-way ANOVA to investigate
    - How MI differs across Java, PHP and Python OSS projects
    - How MI differs across 5 software domains.
  - A reference to average OSS developers with more awareness that the potential options on Languages in terms of maintainability

Future Works

- Other languages, e.g., C/C++, Ruby, JavaScript, etc.
- More language specific factors
  - e.g. programming types, semantics, etc.
- The relationships between maintainability and other OSS quality attributes
  - e.g. how does the maintainability impact on reliability of OSS projects?