West Chester University
Digital Commons @ West Chester University

Computer Science Student Work | Computer Science

4-29-2021

Statistical Analysis of Eclipse Refactoring Bug Reports

Lipika Chandrashekar
West Chester University of Pennsylvania

Jasmine Howard
West Chester University of Pennsylvania

Akash Kumar
West Chester University of Pennsylvania

Eric Lacker
West Chester University of Pennsylvania

Srilaxmi Paramaiahgari
West Chester University of Pennsylvania

Follow this and additional works at: https://digitalcommons.wcupa.edu/compsci_stuwork

Part of the Computer Sciences Commons

Recommended Citation

Chandrashekar, L., Howard, J., Kumar, A., Lacker, E., & Paramaiahgari, S. (2021). Statistical Analysis of Eclipse Refactoring Bug Reports. Retrieved from https://digitalcommons.wcupa.edu/compsci_stuwork/1

This Poster is brought to you for free and open access by the Computer Science at Digital Commons @ West Chester University. It has been accepted for inclusion in Computer Science Student Work by an authorized administrator of Digital Commons @ West Chester University. For more information, please contact wcressler@wcupa.edu.

Statistical Analysis of Eclipse Refactoring Bug Reports

Lipika Chandrashekar, Jasmine Howard, Akash Kumar, Eric Lacker, Srilaxmi Paramaiahgari

Advisor: Dr. Jongwook Kim

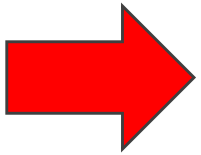
Department of Computer Science

West Chester University

Software Refactoring

- Refactoring is a behavior preserving code transformation
  - e.g., rename

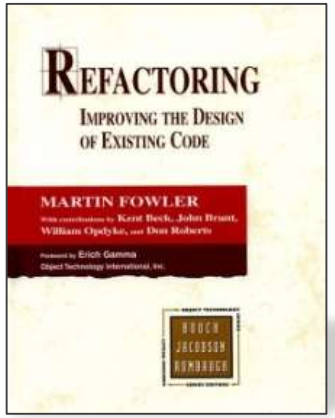
```
class Bank {
    void deposit(int dollars){
        ...
    }

    void addOneDollar(){
        deposit(1);
    }

    void addTenDollars(){
        deposit(10);
    }
    ...
}
```



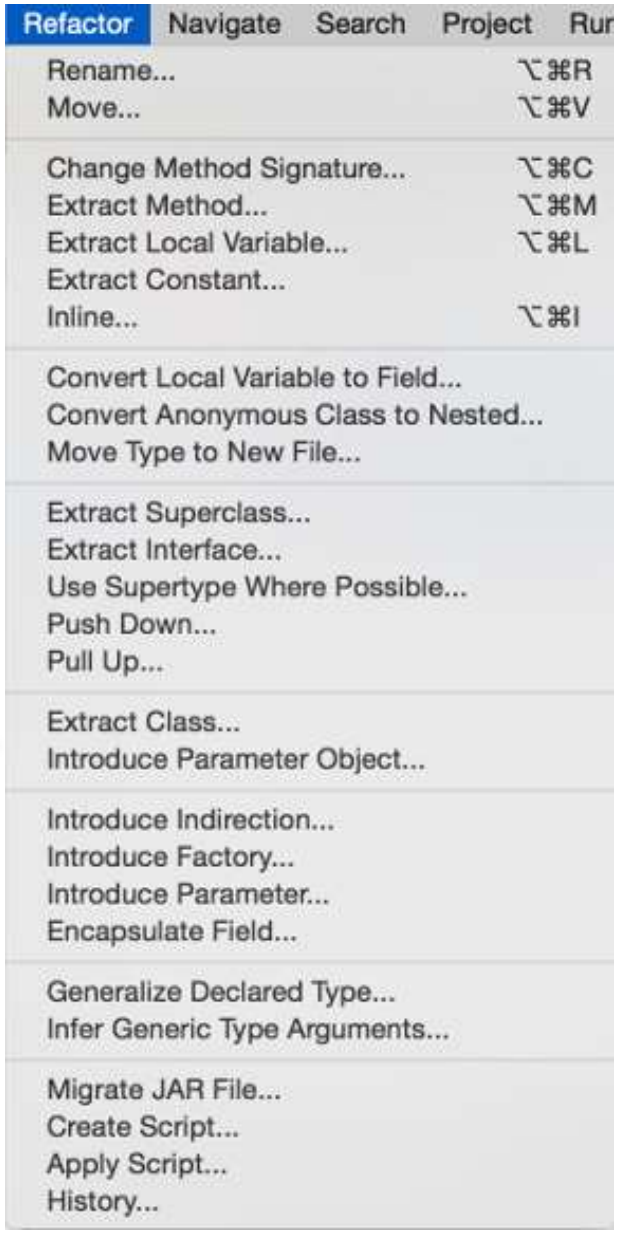
```
class Bank {
    void save(int dollars){
        ...
    }

    void addOneDollar(){
        save(1);
    }

    void addTenDollars(){
        save(10);
    }
    ...
}
```

Refactoring Tools

- Fowler's web site lists over 90 refactorings
  - ~3 refactorings are added each year
  - e.g., rename, move, extract, pull-up, push down, ...



- Today's software tools offer built-in refactorings
  - e.g., Eclipse, NetBeans, IntelliJ IDEA, …



Bug ID	Reporter	Status	Resolution	Summary
385550	Jongwook Kim	RESOLVED	FIXED	[move method] Update reference to inherited field
385989	Jongwook Kim	RESOLVED	FIXED	[move method] Existing type used instead of qualifying the moved one
404471	Jongwook Kim	RESOLVED	FIXED	[move method] Annotation error in applying move-refactoring to inherited methods
404477	Jongwook Kim	RESOLVED	FIXED	[move method] Wrong detection of duplicate methods (can result in compile errors)
404856	Jongwook Kim	CLOSED	DUPLICATE	[move method] Move-refactoring looses comments
411529	Jongwook Kim	RESOLVED	FIXED	[move method refactoring] reference to protected method
416198	Jongwook Kim	RESOLVED	FIXED	[Inline] Inline method refactoring fails to update method reference as of enum constructor call's argument
424388	Jongwook Kim	ASSIGNED	---	[move method] method invocation introduces class-instance-creation as the parameter incorrectly
424654	Jongwook Kim	ASSIGNED	---	[move method] 'null' is assigned to 'this' keyword
425419	Jongwook Kim	CLOSED	DUPLICATE	[Move Field] Field references are not updated
432845	Jongwook Kim	ASSIGNED	---	[move method] incorrect update of references inside anonymous class declaration
434047	Jongwook Kim	ASSIGNED	---	[move method] Moving method with generic method invocation to the generic class results in compiler error
436997	Jongwook Kim	RESOLVED	FIXED	[move method] incorrect precondition of checking references to enclosing instances in generic declaring class.
437233	Jongwook Kim	ASSIGNED	---	[move method] move method cannot detect conflicting methods in parent classes.
437762	Jongwook Kim	RESOLVED	WORK	[move method] incorrect precondition of detecting conflicting methods with super-class-type arguments
439090	Jongwook Kim	ASSIGNED	---	[move method] inherited member has wrong access modifier when referenced in different package
439969	Jongwook Kim	CLOSED	DUPLICATE	[move method] incorrect reference to members of anonymous class declaration
441204	Jongwook Kim	ASSIGNED	---	[move method] wrong precondition check about references to enclosing instances
441217	Jongwook Kim	ASSIGNED	---	[move method] incorrect update of references inherited by anonymous class declaration
444032	Jongwook Kim	ASSIGNED	---	[move method] incorrect update of referencing the member of destination when moved via fields
444284	Jongwook Kim	CLOSED	DUPLICATE	[move method] incorrect update of qualified this expression when moved via fields
448894	Jongwook Kim	ASSIGNED	---	[move method] incorrect variable name which refers to original class
454818	Jongwook Kim	RESOLVED	WORK	[move method] incorrect update of references to a member of generic class declaration
455540	Jongwook Kim	ASSIGNED	---	[pull up] types referenced in pulled-up method change due to missing qualified names.
457987	Jongwook Kim	ASSIGNED	---	[extract method] it should not be allowed to extract if-return statement(s) that return non-void value
458089	Jongwook Kim	ASSIGNED	---	[pull up] incorrect update of the pulled-up member's access modifier
458092	Jongwook Kim	ASSIGNED	---	[pull up] incorrect precondition check on lower visibility of sibling methods
458184	Jongwook Kim	ASSIGNED	---	[pull up] need to a precondition to disallow a method that contains references to non-local type parameters
458464	Jongwook Kim	ASSIGNED	---	[pull up] incorrect replacement of 'super' keyword with 'this' when it references grandparents' member
465069	Jongwook Kim	NEW	---	[move method] incorrect precondition that disallows moving method to enclosing class
465131	Jongwook Kim	NEW	---	[move method] incorrect precondition that misses type parameter capturing
467019	Jongwook Kim	NEW	---	[move method] incorrect precondition that disallows moving methods of anonymous class
467024	Jongwook Kim	ASSIGNED	---	[move method] binding of method call changes due to incomplete conflicting method check
480070	Jongwook Kim	NEW	---	[Push Down] incorrectly reduce visibility of the inherited method
480071	Jongwook Kim	NEW	---	[Push Down] incorrect problem detection about reference(s) to the pushed-down member
490972	Jongwook Kim	NEW	---	[introduce parameter] incorrect parameter replacement to references inside the anonymous class
491541	Jongwook Kim	NEW	---	[pull up] incorrect field when subclasses have fields with different type
495899	Jongwook Kim	NEW	---	[move method] a parameter of the original class type should be introduced
495901	Jongwook Kim	NEW	---	[move method] move method changes the order of evaluating prefix/argument expressions
495902	Jongwook Kim	NEW	---	[change method signature] removing parameter(s) gets rid of argument expressions
495903	Jongwook Kim	NEW	---	[change method signature] adding a new parameter may have a default value with side effects
495905	Jongwook Kim	NEW	---	[change method signature] reordering parameters changes the order of evaluating argument expressions
495907	Jongwook Kim	NEW	---	[move method] move method can reference null-valued prefix (causing null pointer exception)

Eclipse Refactorings are Buggy!

Eclipse Bug Report System

People report bugs using this site

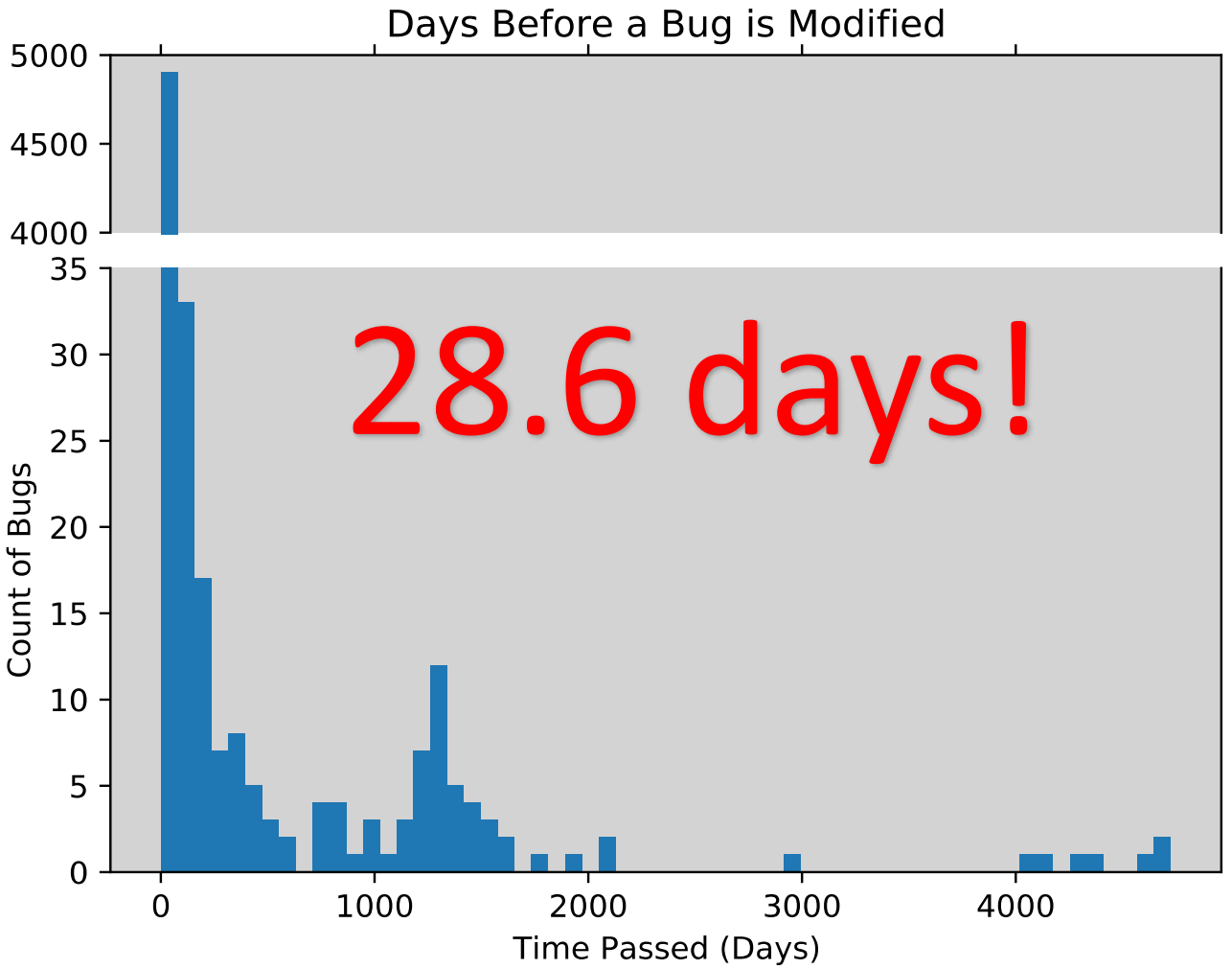
Bugzilla Terms

Status	Resolution
Open Bugs	
UNCONFIRMED CONFIRMED IN_PROGRESS	(None)
Closed Bugs	
RESOLVED	FIXED
	INVALID
VERIFIED	WONTFIX
	DUPLICATE
CLOSED	WORKSFORME

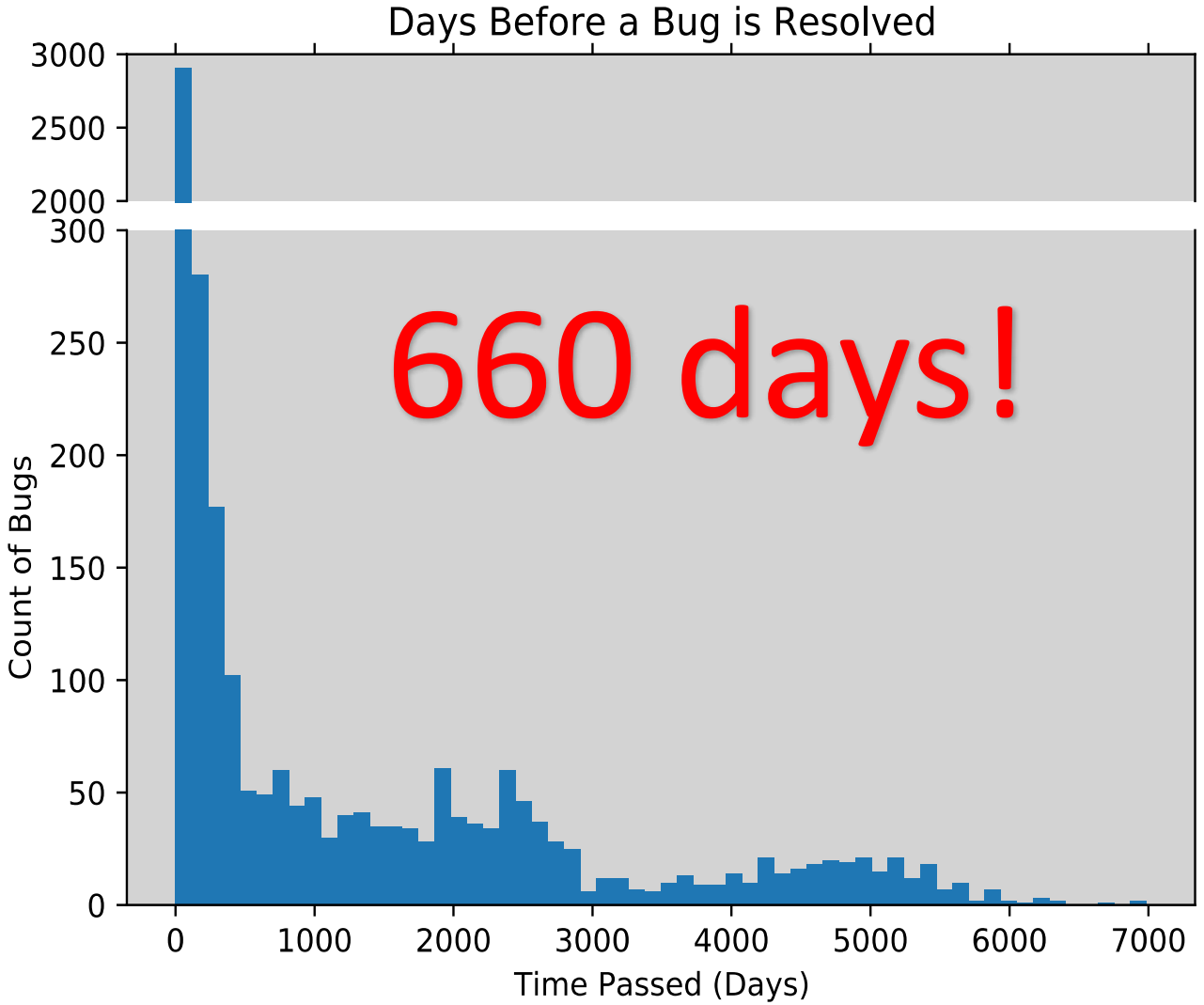
Research Questions

- What is the average time passed before a bug is looked at?
- How long does it take for a refactoring bug to be fixed?
- How many bugs won't be fixed?
- How many duplicate bug reports exist?
- How long a bug has remained in the Bugzilla system?
- …

[RQ #1] Average time passed before a bug is FIRST looked at



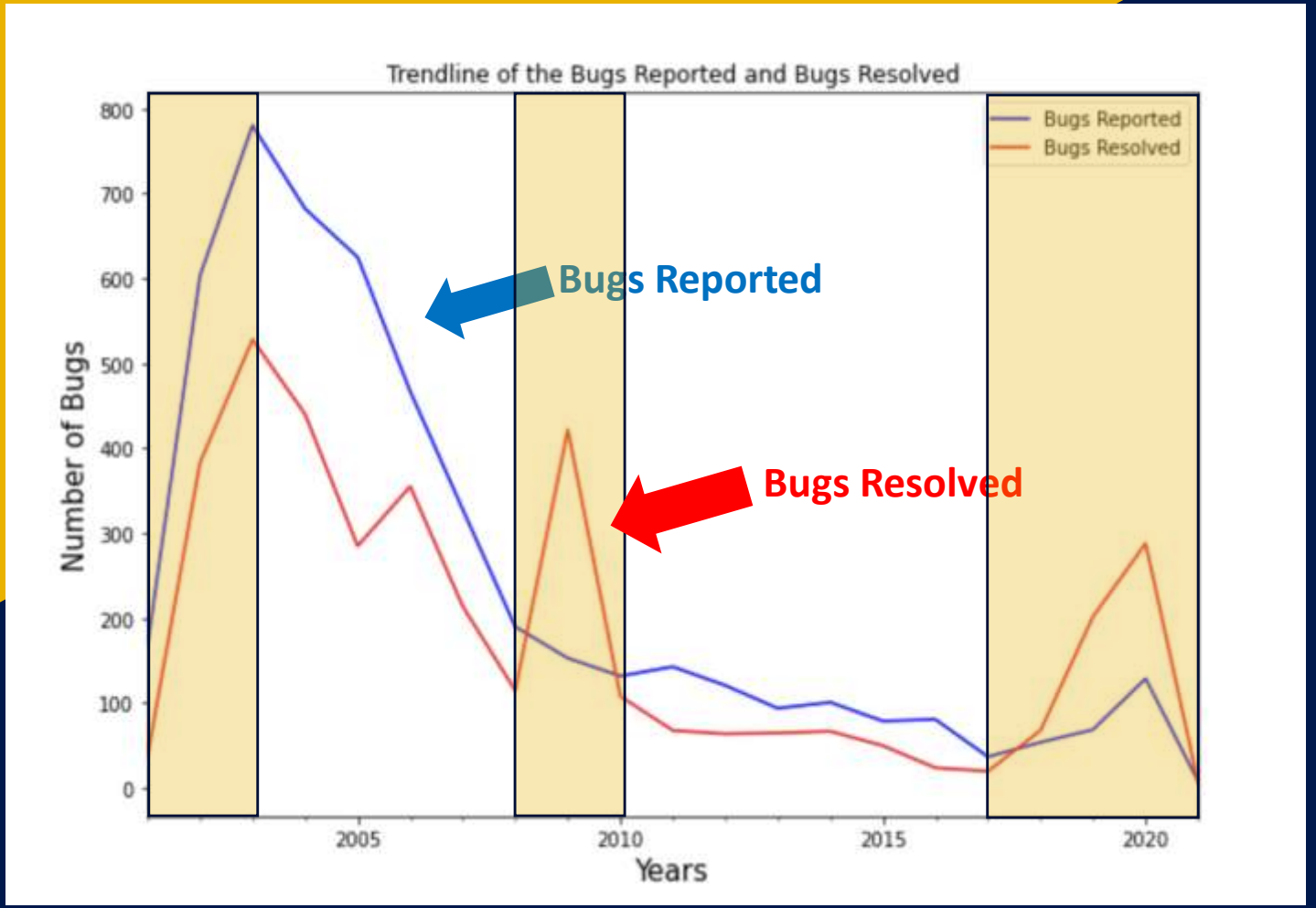
[RQ #2] Average Time taken for a bug to be resolved



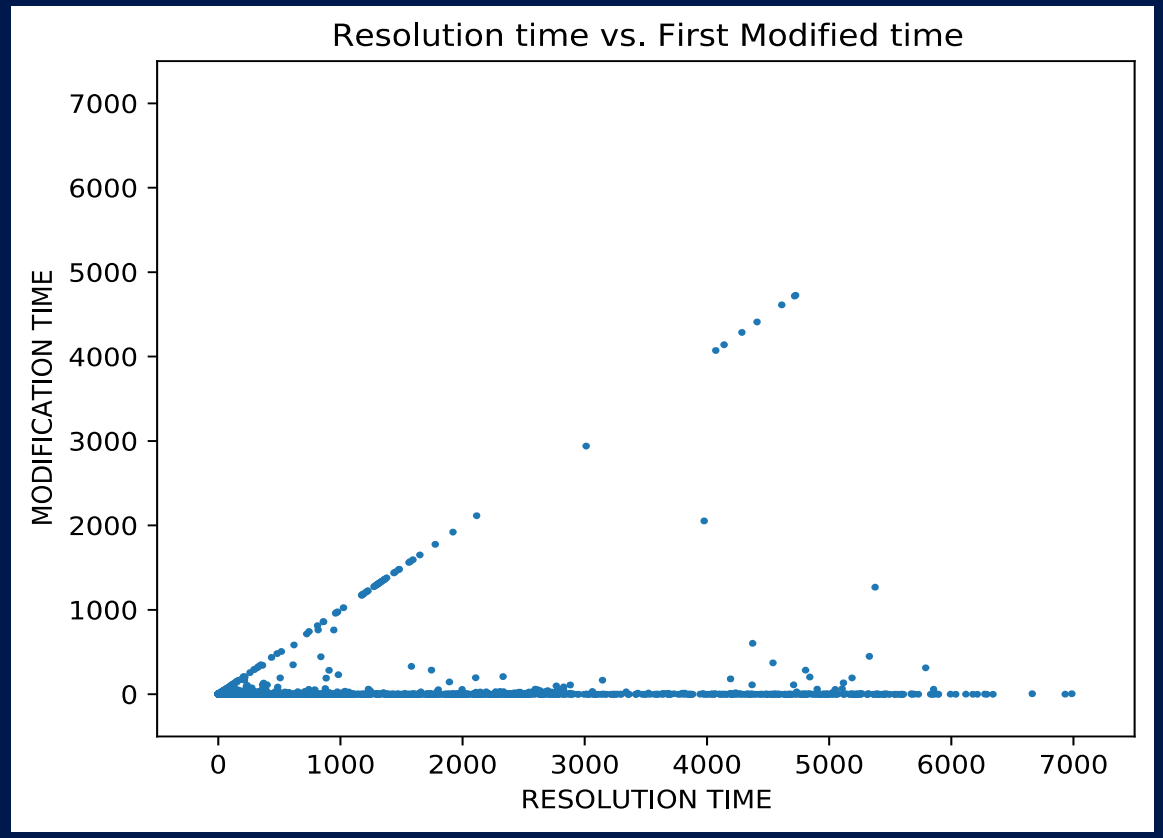
[RQ #3] DUPLICATE and WONTFIX Bugs

Resolutions	DUPLICATE	FIXED	INVALID	WONTFIX	WORKSFORME	Total
Frequency	742	2423	139	929	423	5047
Percent	14.7	48.1	2.8	18.4	8.4	100

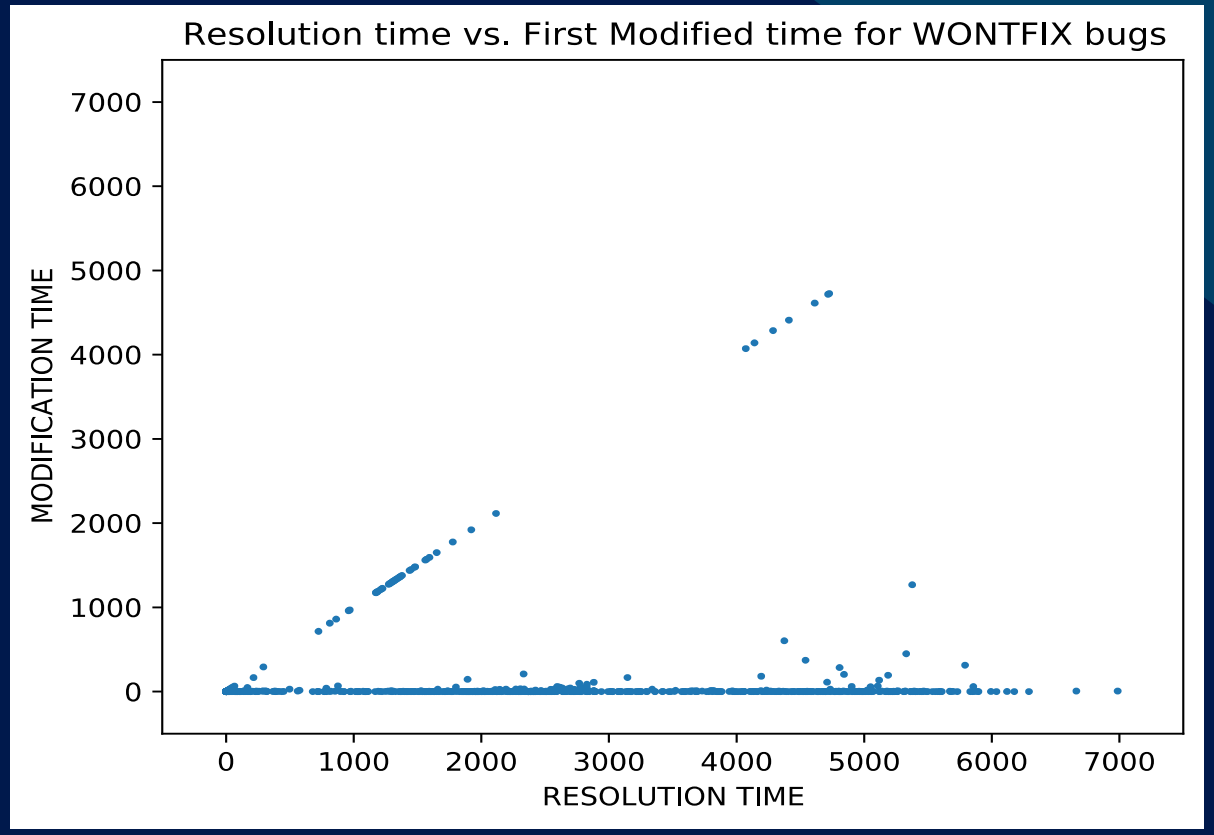
[RQ #4] Trend of the Bug Report/Resolution over Years



[RQ #5] Resolution Time vs First Modified Time

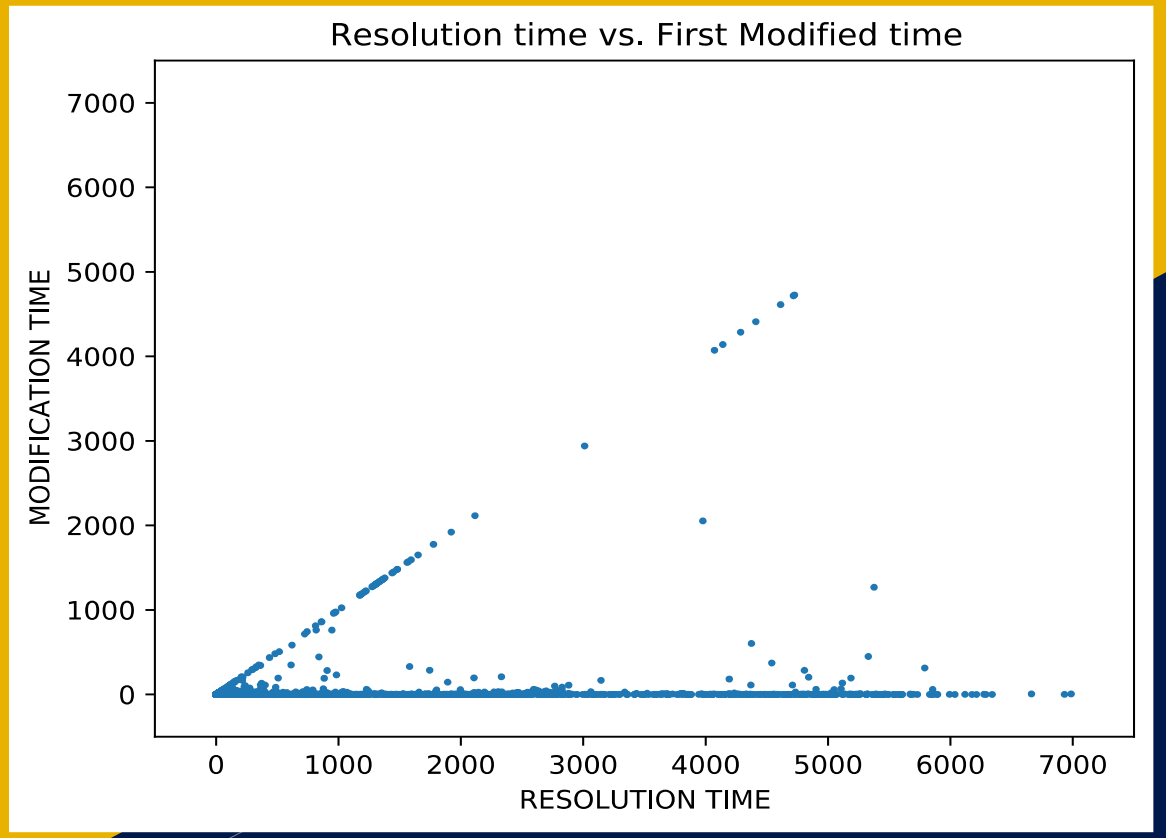


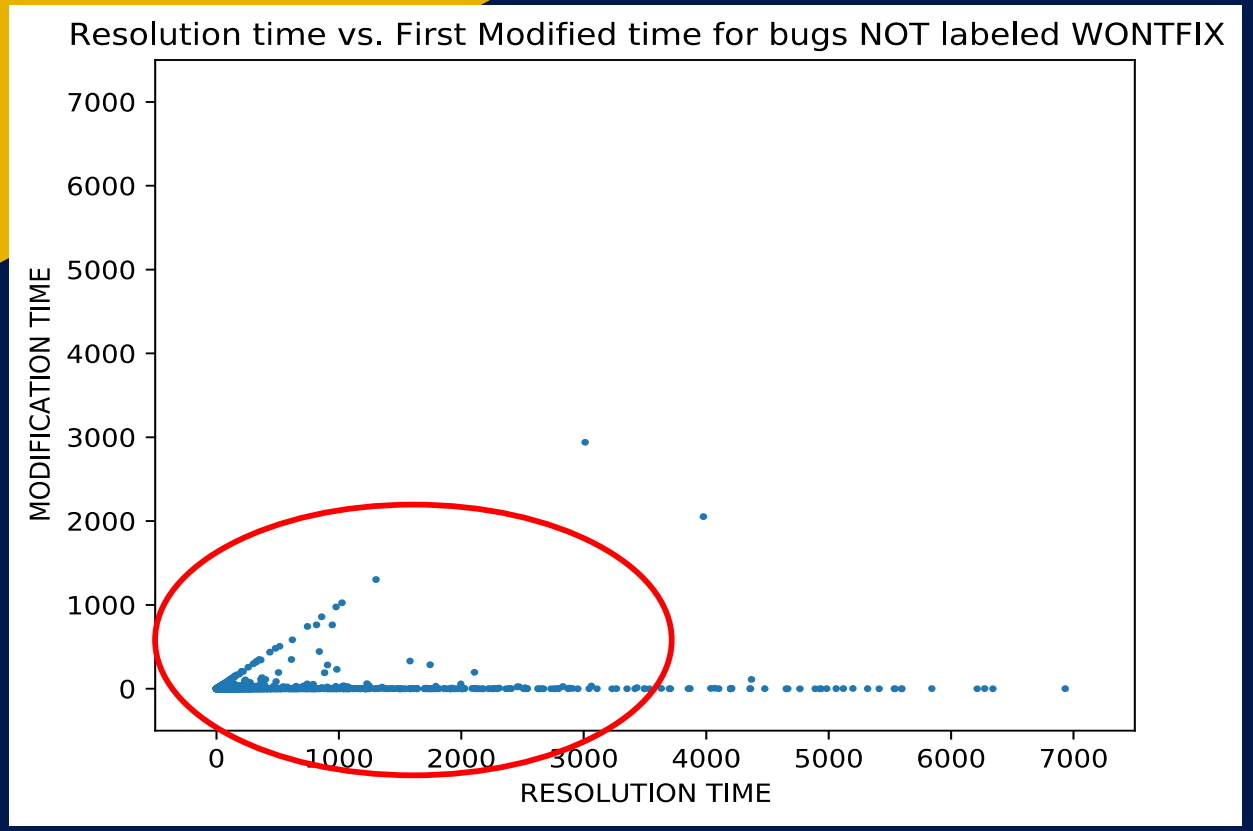
All Bugs



WONTFIX Bugs

[RQ #6] Resolution Time vs First Modified Time Excluding WONTFIX Bugs

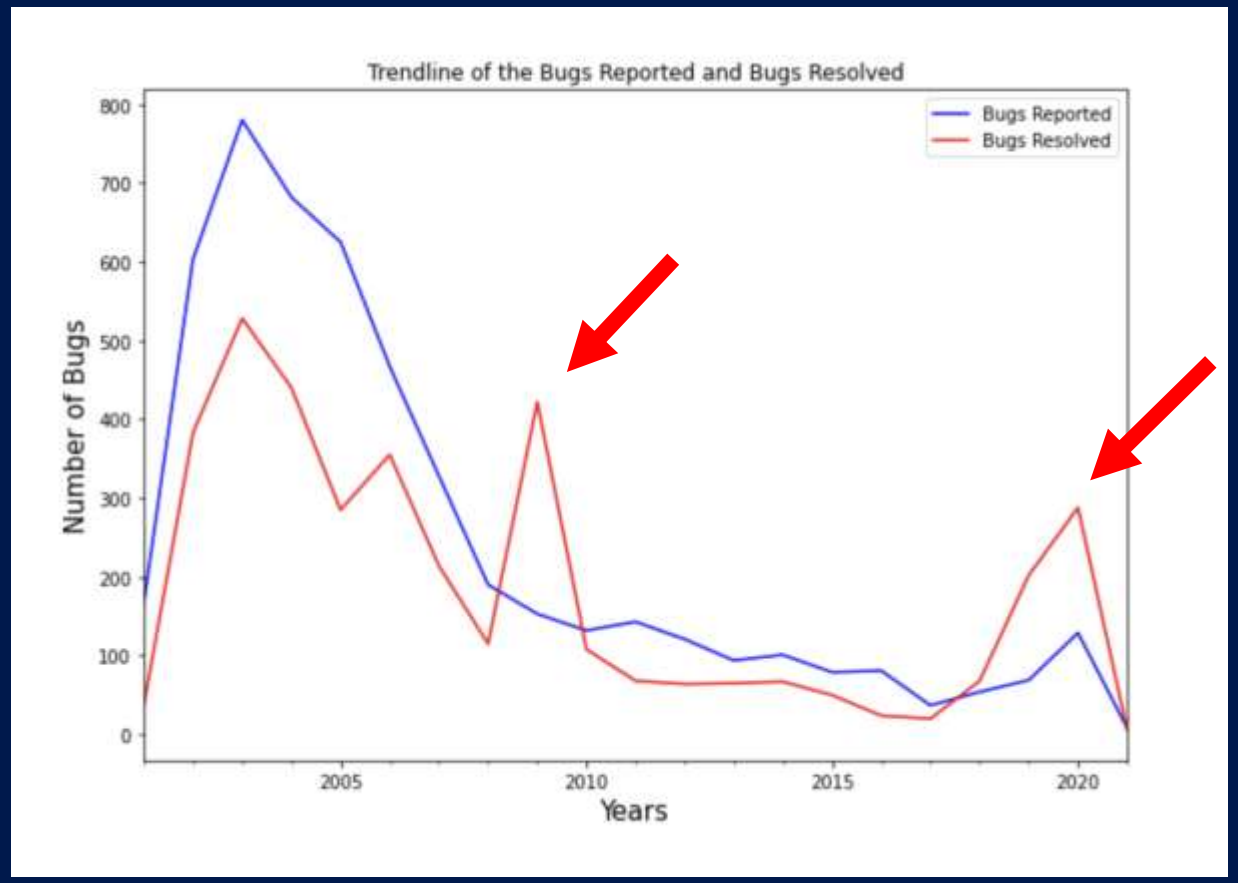


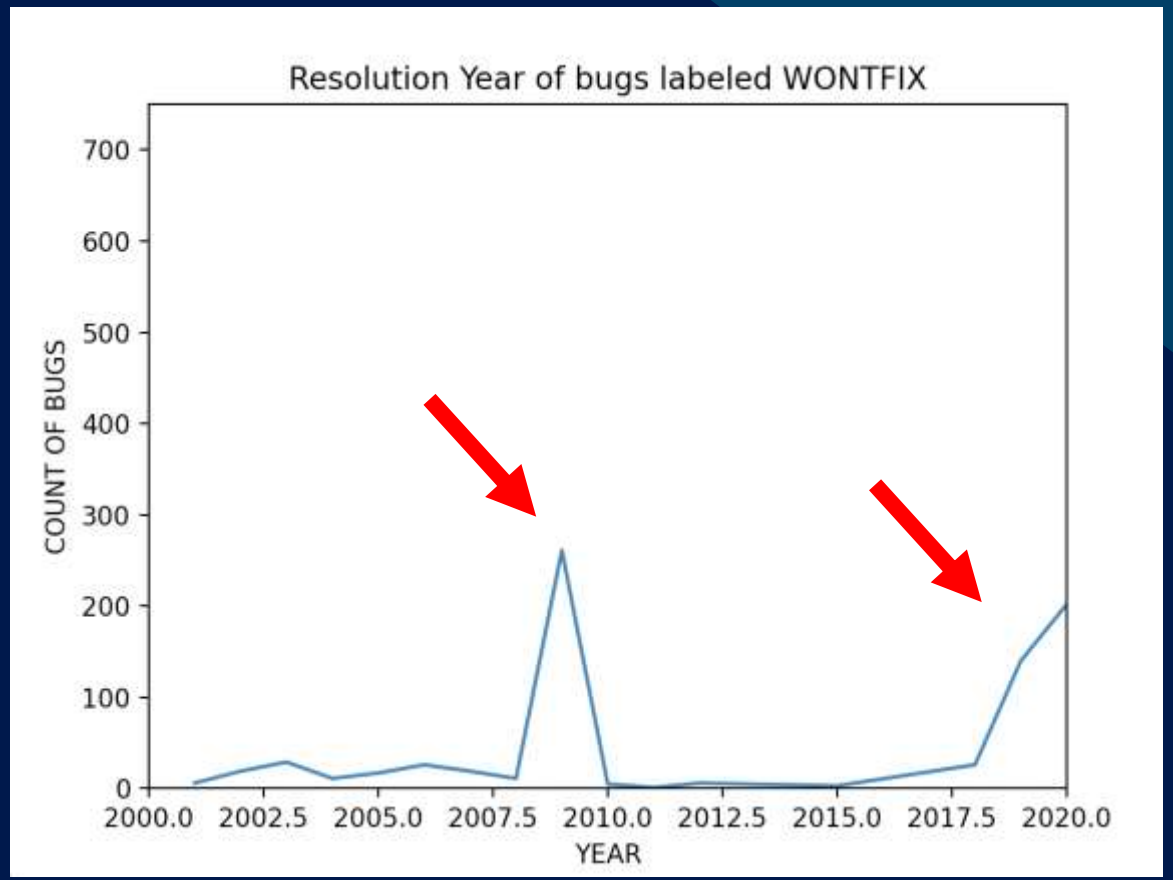


All Bugs

Excluding WONTFIX Bugs

[RQ #7] WONTFIX vs. Resolution Trendline Comparison





Reasons for the spikes in 2009 and 2019-2020

Conclusion

- Eclipse Refactoring does not always **preserve** the original behavior like expected
- On average, it took about **a month** for a bug to be first looked at
- Average time taken to resolve a bug was **660 days**, almost two years!
- **20% bugs won’t be fixed** and 15% are duplicates
- Eclipse IDE has plenty of room for improvement!

Any Questions?