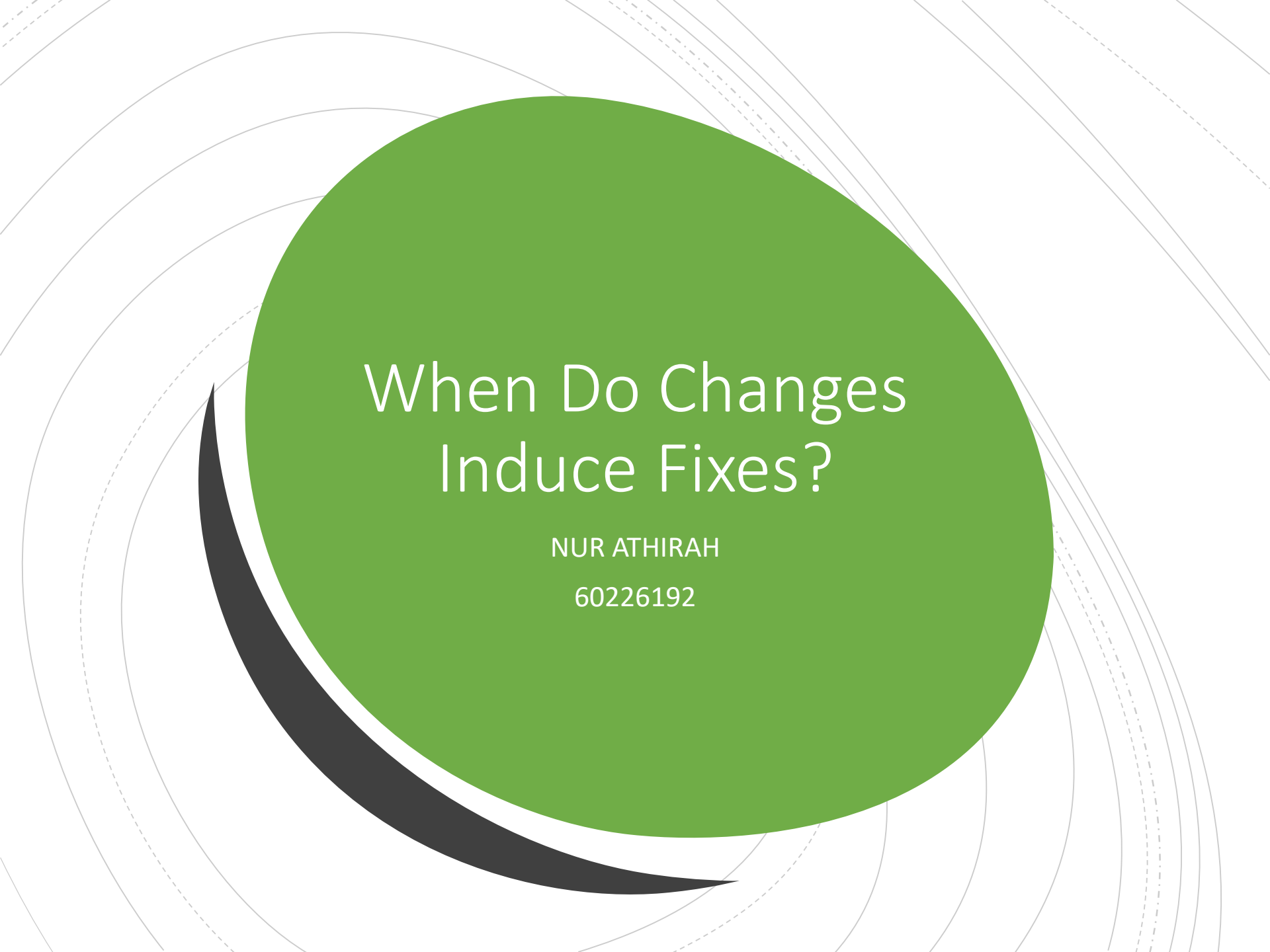


The background features a series of concentric circles in light gray, some solid and some dashed. A large, solid green circle is positioned in the center, containing the title and author information. A dark gray, curved, comma-like shape is located to the left of the green circle.

# When Do Changes Induce Fixes?

NUR ATHIRAH

60226192

# AIM

- attempt to identify those changes that caused problems
- the basic idea is as follows:
  1. start with a bug report in the bug database, indicating a fixed problem
  2. extract the associated change from the version archive, thus giving us the location of the fix
  3. determine the earlier change at this location that was applied before the bug was reported

# BACKGROUND

- When we mine software histories, we frequently do so in order to detect patterns that help us understanding the current state of the system.
- Unfortunately, not all changes in the past have been beneficial. Any bug database will show a significant fraction of problems that are reported some time after some change has been made.

# WHAT IS FIX INDUCING?

- This earlier change is the one that *caused* the later fix

# What can one do with fix-inducing changes?

Here are some potential applications:

- Which change properties may lead to problems?
- How error-prone is my product?
- How can I filter out problematic changes?
- Can I improve guidance along related changes?

# METHOD



EXTRACT DATA FROM  
VERSION AND BUG  
ARCHIVES



LINK BUG REPORTS TO  
CHANGES

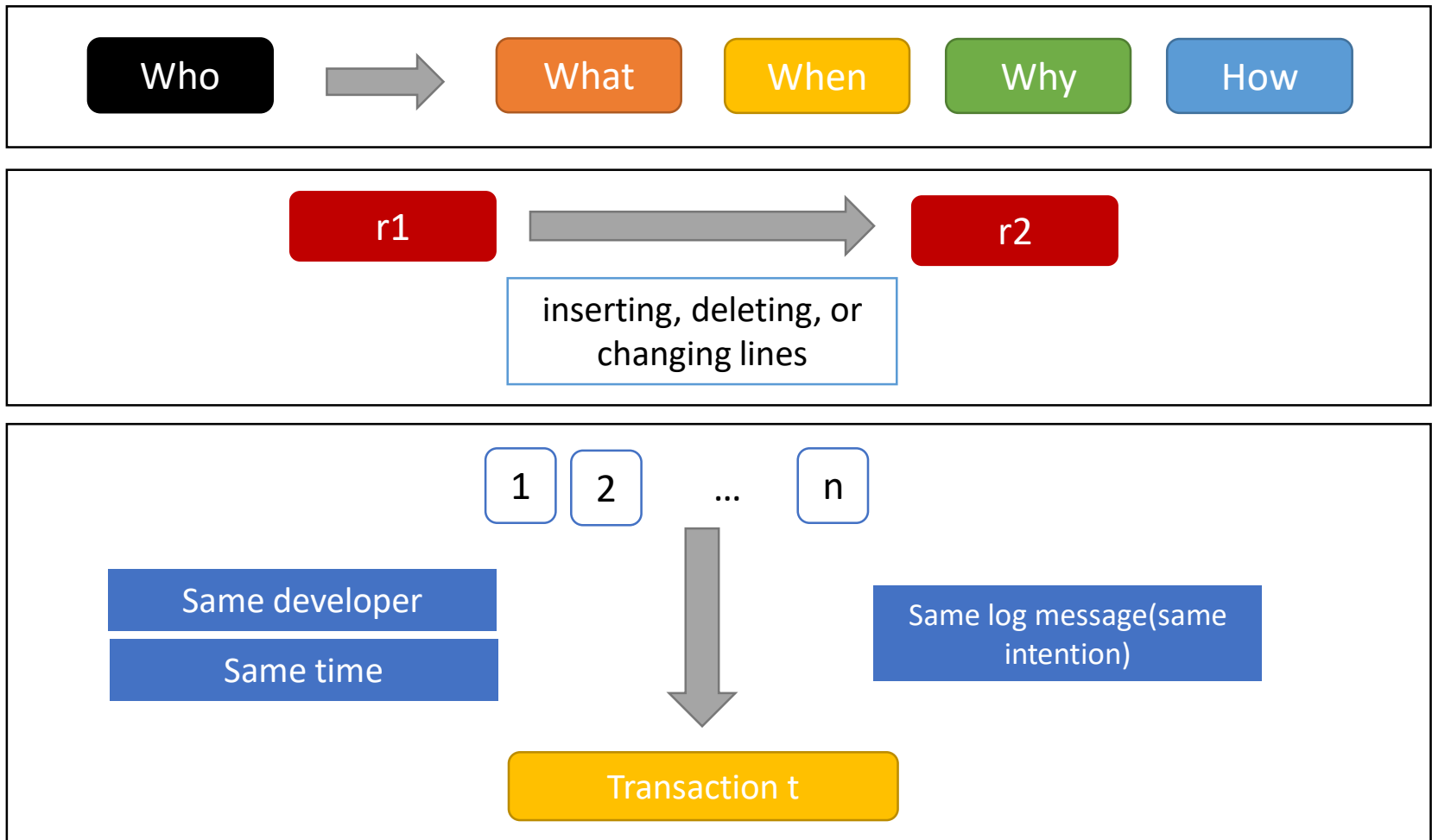


IDENTIFY AND LOCATE  
FIX-INDUCING CHANGES

# How to extract data from version and bug archives

- need all changes and all fixes of a project
- the data(changes and fixes of a project) for the analysis is taken from version archives like CVS and bug tracking systems like BUGZILLA

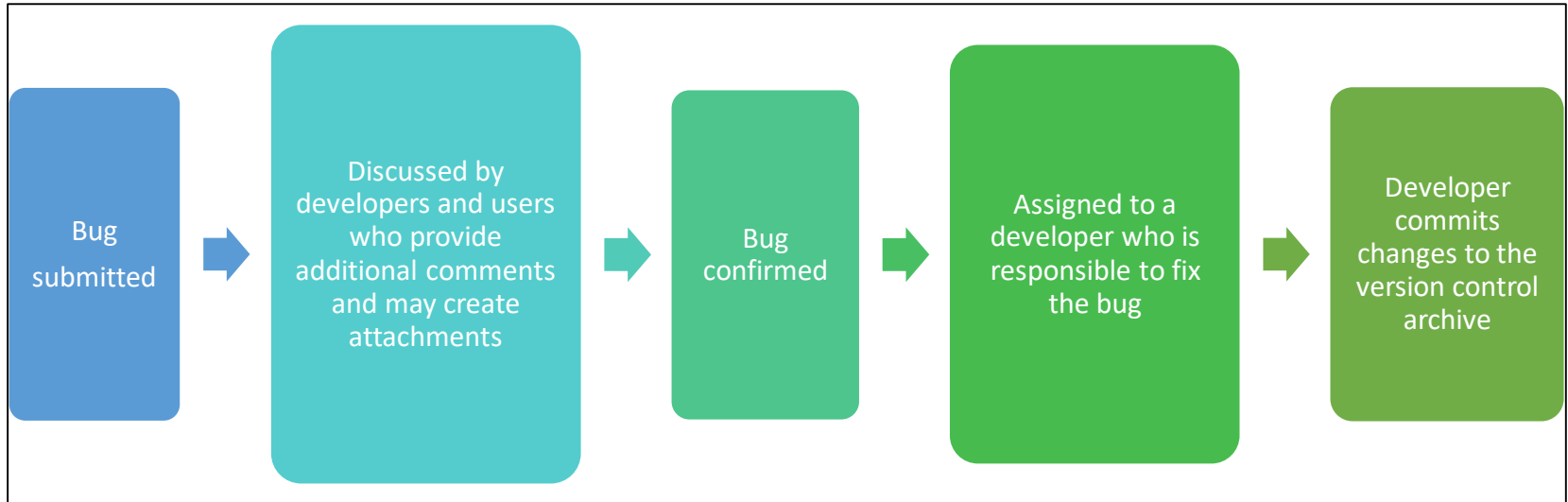
# CVS archive



lacks information about the purpose of a change

# BUGZILLA

- collects bug reports that are submitted by a reporter with a short description and a summary



| Status                                                                                                                             | Resolution                                                                                    |
|------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|
| <ul style="list-style-type: none"><li>• UNCONFIRMED</li><li>• NEW</li><li>• ASSIGNED</li><li>• RESOLVED</li><li>• CLOSED</li></ul> | <ul style="list-style-type: none"><li>• FIXED</li><li>• DUPLICATE</li><li>• INVALID</li></ul> |

Details on the lifecycle of a bug can be found in the BUGZILLA documentation

Link bug reports to changes

# IDENTIFYING FIXES

- assigning every link  $(t, b)$  between a transaction  $t$  and a bug  $b$  two independent levels of confidence:
  - a syntactic level, inferring links from a CVS log to a bug report
  - a semantic level, validating a link via the bug report data
- These levels are later used to decide which links shall be taken into account in our experiments

# SYNTACTIC ANALYSIS

- every log message are divided into stream of tokens
- a token is one of the following
  - a bug number
  - a plain number
  - a keyword
  - a word
- For each link, we initially assign a syntactic confidence  $syn$  of zero and raise the confidence by one for each of the following conditions that is met:
  1. The number is a bug number.
  2. The log message contains a keyword, or the log message contains only plain or bug numbers.

# EXAMPLE

- The syntactic confidence always an integer number between 0 and 2.
- As an example, consider the following log messages:
- Fixed bug 53784: .class file missing from jar file export
  - The link to the bug number 53784 gets a syntactic confidence of 2 because it matches the regular expression for bug and contains the keyword fixed.
- 52264, 51529
  - The links to bugs 52264 and 51529 have syntactic confidence 1 because the log message contains only numbers.
- Updated copyrights to 2004
  - The link to the bug number 2004 has a syntactic confidence of 0 because there is no syntactic evidence that this number refers to a bug.

# SEMANTIC ANALYSIS

- Initially, a link (t, b) has semantic confidence of 0 which is raised by 1 whenever one of the following conditions is met:
  - The bug b has been resolved as FIXED at least once
  - The short description of the bug report b is contained in the log message of the transaction t
  - The author of the transaction t has been assigned to the bug b
  - One or more of the files affected by the transaction t have been attached to the bug b

# EXAMPLE

- Consider the following examples from ECLIPSE, which all have low confidence levels:
- Updated copyrights to 2004
  - The potential bug report number “2004” is marked as invalid and thus the semantic confidence of the link is zero.
- Fixed bug mentioned in bug 64129, comment 6
  - The number “6” appears in the comment for a fix. The syntactic confidence is 1, but the semantic confidence is 0.
- Support expression like `(i) += 3;` and `new int[] {1}[0] +` syntax error improvement
  - “1” and “3” are (mistakenly) interpreted as bug report numbers here. Since the bug reports 1 and 3 have been fixed, the links both get a semantic confidence of 1.

# ANALYSIS RESULT

| syn / sem | 0        | 1           | 2            | 3           | 4      | total         |
|-----------|----------|-------------|--------------|-------------|--------|---------------|
| 0         | 270 (1%) | 1,287 (5%)  | 2,057 (8%)   | 1,439 (6%)  | 2 (0%) | 5,055 (20%)   |
| 1         | 324 (1%) | 4,152 (16%) | 9,265 (37%)  | 1,581 (6%)  | 5 (0%) | 15,327 (61%)  |
| 2         | 110 (0%) | 1,922 (8%)  | 2,421 (10%)  | 482 (2%)    | 0 (0%) | 4,935 (19%)   |
| total     | 704 (3%) | 7,361 (29%) | 13,743 (54%) | 3,502 (14%) | 7 (0%) | 25,317 (100%) |

**Table 1: Distribution of links across different classes of syntactic and semantic confidence levels in ECLIPSE**

| syn / sem | 0          | 1            | 2            | 3          | 4       | total         |
|-----------|------------|--------------|--------------|------------|---------|---------------|
| 0         | 560 (1%)   | 2,899 (5%)   | 4,281 (8%)   | 639 (1%)   | 8 (0%)  | 8,387 (16%)   |
| 1         | 1,211 (2%) | 9,059 (17%)  | 16,336 (30%) | 2,241 (4%) | 22 (0%) | 28,669 (54%)  |
| 2         | 478 (1%)   | 5,250 (10%)  | 9,133 (17%)  | 1,645 (3%) | 12 (0%) | 16,518 (31%)  |
| total     | 2,249 (4%) | 17,208 (32%) | 29,750 (55%) | 4,525 (8%) | 42 (0%) | 53,574 (100%) |

**Table 2: Distribution of links across different classes of syntactic and semantic confidence levels in MOZILLA**

- We identified 25,317 links for ECLIPSE and 53,574 links for MOZILLA
- Based on a manual inspection of several randomly chosen links, we decided to use only those links whose syntactic and semantic levels of confidence satisfy the following condition:

$$\text{sem} > 1 \vee (\text{sem} = 1 \wedge \text{syn} > 0)$$

- Notice that we disregard less than 10% of links for both projects

# How to identify and locate fix-inducing changes

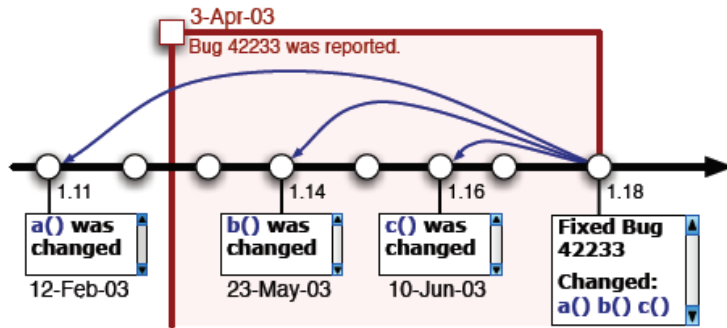


Figure 2: Locate fix-inducing changes for bug 42233

```
$ cvs annotate -r 1.17 Foo.java
...
19:1.11 (john 12-Feb-03): public int a() {
20:1.11 (john 12-Feb-03):     return i/0;
...
39:1.10 (mary 12-Jan-03): public int b() {
40:1.14 (kate 23-May-03):     return 42;
...
59:1.10 (mary 17-Jan-03): public void c() {
60:1.16 (mary 10-Jun-03):     int i=0;
...
```

Figure 3: CVS annotations for Foo.java

- Diff command – to locate fixes
- Annotate command – to display the most recent revision, the developer and the date
- a suspect (ra, rb) is a partial fix if ra is a fix
- a suspect (ra, rb) is a weak suspect if there exists a pair (ra, rc) which is not a suspect
- a suspect (ra, rb) is a hard suspect if it is neither a partial fix, nor a weak suspect
  - A hard suspect indicates a revision for which there is no real evidence of being fix-inducing
- a revision r is fix-inducing if there exists a pair (r, rx) ∈ S which is not a hard suspect
- a transaction t is fix-inducing if one of its revisions is fix-inducing

# MAIN RESULT

# RESULT

- extracted fix-inducing changes for two large open-source projects: ECLIPSE and MOZILLA
- considered all changes and bugs until January 20, 2005;
- database contains
  - 78,954 transactions for ECLIPSE
  - 109,658 transactions for MOZILLA
- They account for 278,010 and 392,972 individual revisions for both projects, respectively.

# Fix-Inducing Transactions are Large

|            | fix-inducing      | $\neg$ fix-inducing | all              |
|------------|-------------------|---------------------|------------------|
| fix        | $3.82 \pm 26.32$  | $2.08 \pm 7.42$     | $2.73 \pm 7.87$  |
| $\neg$ fix | $11.30 \pm 63.02$ | $2.77 \pm 14.94$    | $3.81 \pm 26.32$ |
| all        | $7.49 \pm 44.37$  | $2.61 \pm 13.66$    | $3.52 \pm 22.81$ |

Table 3: Average sizes of fix and fix-inducing transactions for ECLIPSE

|            | fix-inducing     | $\neg$ fix-inducing | all              |
|------------|------------------|---------------------|------------------|
| fix        | $5.79 \pm 37.37$ | $2.12 \pm 9.74$     | $4.39 \pm 30.05$ |
| $\neg$ fix | $4.61 \pm 30.59$ | $1.91 \pm 10.30$    | $3.05 \pm 21.39$ |
| all        | $5.19 \pm 34.12$ | $1.97 \pm 10.13$    | $3.58 \pm 25.23$ |

Table 4: Average sizes of fix and fix-inducing transactions for MOZILLA

- examined if the span of the transaction correlates with the fact that the transaction is fix-inducing
- The transactions are split into four classes, depending on whether the transaction is a fix, fix-inducing, both, or none
- Table 3: fix-inducing transactions are roughly three times larger than non fix-inducing transactions
- Table 4: MOZILLA which shows a similar trend

# Don't Program on Fridays

| % of revisions                            | Day of Week |      |      |      |      |      |      | avg  |
|-------------------------------------------|-------------|------|------|------|------|------|------|------|
|                                           | Mon         | Tue  | Wed  | Thu  | Fri  | Sat  | Sun  |      |
| $P(\text{fix})$                           | 18.4        | 20.9 | 20.0 | 22.3 | 24.0 | 14.7 | 16.9 | 20.8 |
| $P(\text{bug})$                           | 11.3        | 10.4 | 11.1 | 12.1 | 12.2 | 11.7 | 11.6 | 11.4 |
| $P(\text{bug} \cap \text{fix})$           | 4.6         | 4.8  | 4.6  | 5.2  | 5.6  | 4.5  | 4.5  | 4.9  |
| $P(\neg \text{bug} \cap \neg \text{fix})$ | 74.9        | 73.5 | 73.5 | 70.8 | 63.4 | 78.1 | 76.0 | 72.7 |
| $P(\text{bug}   \text{fix})$              | 25.1        | 22.9 | 23.3 | 23.5 | 23.2 | 30.3 | 26.4 | 23.7 |
| $P(\text{bug}   \neg \text{fix})$         | 8.2         | 7.1  | 8.1  | 8.8  | 8.7  | 8.4  | 8.6  | 8.1  |

Table 5: Distribution of fixes and fix-inducing changes across day of week in ECLIPSE

| % of revisions                            | Day of Week |      |      |      |      |      |      | avg  |
|-------------------------------------------|-------------|------|------|------|------|------|------|------|
|                                           | Mon         | Tue  | Wed  | Thu  | Fri  | Sat  | Sun  |      |
| $P(\text{fix})$                           | 42.5        | 46.5 | 49.7 | 45.9 | 48.4 | 50.2 | 61.1 | 48.5 |
| $P(\text{bug})$                           | 39.1        | 44.1 | 41.2 | 40.8 | 46.2 | 44.9 | 26.4 | 41.5 |
| $P(\text{bug} \cap \text{fix})$           | 19.4        | 23.6 | 22.8 | 21.6 | 26.9 | 19.6 | 13.2 | 21.9 |
| $P(\neg \text{bug} \cap \neg \text{fix})$ | 37.8        | 33.0 | 31.9 | 34.9 | 32.3 | 24.5 | 25.7 | 31.9 |
| $P(\text{bug}   \text{fix})$              | 45.7        | 50.8 | 45.8 | 47.1 | 55.6 | 39.1 | 21.6 | 45.2 |
| $P(\text{bug}   \neg \text{fix})$         | 34.1        | 38.3 | 36.7 | 35.5 | 37.3 | 50.6 | 33.9 | 38.1 |

Table 6: Distribution of fixes and fix-inducing changes across day of week in MOZILLA

- broke down changes by the day of the week when they were applied
- $P(\text{fix})$  - fixes were done
  - ECLIPSE: Friday
  - MOZILLA: Sunday
- The risk that a fix will be later undone is highest for
  - ECLIPSE: Saturday
  - MOZILLA: Friday
- $P(\text{bug} | \text{fix})$  – fix that has bug
  - ECLIPSE: a fix-inducing change is almost three times higher for fixes
  - MOZILLA: does not hold the same result
- $P(\text{bug} | \neg \text{fix})$  – change that has bug

# CONCLUSION

- As soon as a project has a bug database as well as a version archive, we can link the two to identify those changes that caused a problem.
- Such fix-inducing changes have a wide range of applications.(can be used for many other usages)
- In this paper, we examined the properties of fix-inducing changes in the ECLIPSE and MOZILLA projects and found, among others, that
  - the larger a change, the more likely it is to induce a fix;
- We also found that in the ECLIPSE project, fixes are three times as likely to induce a later change