Problem Solving

SWE 795, Fall 2019
Software Engineering Environments
Today

- Part 1 (Lecture)(~75 mins)
- Break!

- Part 2 (Discussion)(45 mins)
  - Discussion of readings

- Part 3 (Project group work)(30 mins)
  - Time to work in groups, ask questions
Logistics

• HW1 due next week

• No readings next week, will have project presentations instead
A few minutes in the life of a developer
A few hours in the life of a professional software developer

| collaboration | Developer assigned bug by team |
| programming   | Reproduces error |
|              | Browser hits error message (500 internal error) |
|              | Attaches debugger |
|              | Browse to page again, hit null reference exception |
|              | Hypothesize from call stack which function might be responsible |
|              | Browse through code |
|              | Uses debugger to change values & experiment |
|              | Make change, recompile, check, doesn’t work |
|              | Navigates slice, wrong values came from objects |
|              | In complicated code doesn’t understand |
| collaboration | Walks to B’s office and asks where data comes from |
|              | B working on high profile feature in area |
| programming   | Tries to make change, still doesn’t work |
| collaboration | Walks back to B, realize related to C’s feature, C at lunch |
|              | After lunch, A and B walk to C’s office, |
| design        | A, B, C change design to work with new feature |
| collaboration | Bug passed from A to C to change feature |
**Problem solving**

**Goal**: where am I trying to go?

**Operators**: what actions can I take to get closer to the goal?

Apply operator, look at new state, apply another operator

<table>
<thead>
<tr>
<th>task</th>
<th>Investigate and fix a design problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>question</td>
<td>Why is an event being issued by forcing a cache update?</td>
</tr>
<tr>
<td>action</td>
<td>How is BufferHandler using its buffer field? Are there any other mutations on it?</td>
</tr>
<tr>
<td>action</td>
<td>Read methods of BufferHandler</td>
</tr>
<tr>
<td>action</td>
<td>Why is there a buffer member variable that is never used?</td>
</tr>
<tr>
<td>action</td>
<td>Investigate references to BufferHandler.buffer</td>
</tr>
<tr>
<td>action</td>
<td>Why is doDelayedUpdate() a member of BufferHandler?</td>
</tr>
<tr>
<td>action</td>
<td>Reads methods along path, concludes that BufferHandler tracks update delays</td>
</tr>
<tr>
<td>action</td>
<td>Why wouldn’t isFoldStart() call getFoldLevel()</td>
</tr>
<tr>
<td>action</td>
<td>Reads isFoldStart(), getFoldAtLine()</td>
</tr>
<tr>
<td>action</td>
<td>Concludes isFoldStart() doesn’t call because of short circuit evaluation</td>
</tr>
<tr>
<td>action</td>
<td>Implement fix</td>
</tr>
<tr>
<td>action</td>
<td>Assure correctness</td>
</tr>
<tr>
<td>action</td>
<td>Set conditional break point</td>
</tr>
<tr>
<td>action</td>
<td>Check that jEdit still appears to work correctly</td>
</tr>
<tr>
<td>action</td>
<td>Repro original bug by reinserting</td>
</tr>
</tbody>
</table>
Problem solving is recursive
Problem solving involves answering questions

Goal: Fix issue

- Where is the defect?
- Which function generated the incorrect output?
- Where is this function invoked?

Subgoal
Problem solving involves strategies

Goal: Fix issue

- Where is the defect?
- Trace output backwards

Subgoal

Subgoal

Subgoal

Subgoal

Subgoal

Which function generated the incorrect output?
Problem solving involves taking actions to answer questions and follow strategies.

Goal: Fix issue

Where is the defect?

Trace output backwards

Which function generated the incorrect output?
Developers use a variety of techniques for obtaining information and answering questions.
Problem solving involves formulating hypotheses

Goal:
Fix issue

Where is the defect?

Maybe it's in the registration code I just wrote

Is the registration code executing correctly?
Problem solving involves choices between strategies

Goal: Fix issue

Where is the defect?

Maybe it's in the registration code I just wrote

Is the registration code executing correctly?

Trace output backwards

Which function generated the incorrect output?
Problem solving in programming

• Developers have **tasks** (e.g., fix this defect, implement this feature) which they work to complete.

• Developers ask **questions** reflecting information they need in order to complete tasks.
  - e.g., What’s the best design for implementing this?

• Developers may formulate **hypotheses** representing answers to questions.

• Developers select **strategies** to gather evidence answer questions and to support or reject hypotheses.
  - From code, from external resources, from teammates

• Developers often have multiple strategies to answer questions.
LaToza, Garlan, Herbsleb, Myers. Program comprehension as fact finding. FSE 07.
Supporting programming activities

Goal: Fix issue

Where is the defect?

Maybe it's in the registration code I just wrote

Which function generated the incorrect output?

Is the registration code executing

Trace output backwards

• Many potential points of intervention, supporting subgoals / strategies / question answering / testing hypotheses
Useful interventions solve important problems

LaToza and Myers. Designing useful tools for developers. PLATEAU 2011.
What percentage of the last week have you spent...

Example: Activities in fixing a defect

Circle size: % of time
Edge thickness: % of transitions observed

For tasks in code in your own codebase that you haven’t seen recently

LaToza and Myers. Developers ask reachability questions. ICSE 2010.
Figure 2. The backgrounds and task structures of the 17 observed developers. The right edge of each task block indicates the reason for the task switch (thin line for done, thick line for blocked, jagged line for interrupted). When a task gets broken up by interruptions or blocking, we draw its fragments at the same vertical level to show resumption.
Some methods for supporting problem solving

• Find an important question, build tool that makes it easier to answer

• Find an action that helps developers answer questions, make it easier to take

• Find a new strategy that helps developers answer question more effectively
Many other factors influence difficulty answering questions

- expertise
- development environments
- programming languages
- code quality
- team practices

• Interventions might also target these factors

- time to market
- software quality
Some methods for supporting problem solving

• Find an important question, build tool that makes it easier to answer

• Find an action that helps developers answer questions, make it easier to take

• Find a new strategy that helps developers answer question more effectively
Making questions easier to answer

• Tools help developers be more productive by reducing the time to answer questions, increasing likelihood of success

• This requires
  • understanding **precisely** the information required and context available to developers
  • insight into a **mechanism** to make a question easier to answer
Example: Questions about object structure

Is a
Who implements type X? [who can be an object or a type]

Navigability
Let's say I am in the StandardDrawing class and I want the JavaDrawApp object which is a DrawingEditor [...] What would save me a lot of time is to say now I am at the Drawing and I want to go to the DrawingEditor, show me my options.

Part of
Maybe I would start with the Drawing object and that should have a list of listeners?

How to get
How I will get hold of the DrawingEditor object? [...] Basically I need to know the instance of the current window.

Is in tier
I know I need to get the view from here; so how do I do that?

Is owned
What I would be interested in is looking in the code to try to understand where are the view and model

Cardinality
The class diagram says that the DrawingEditor has one DrawingView and the StandardDrawingView may or may not have a Drawing.

May alias
I would like to know the cardinality: so Window has one or more StandardDrawingViews?

Has a
Maybe I would start with the Drawing object and that should have a list of listeners

Cardinality
[...] the window itself has a reference to the UndoManager but you can’t tell from this diagram whether each window has its own UndoManager, or whether it is just one global manager.

Points to
So I have different selections in the different views.

Both of them are two views on the same Drawing, but if there are two windows...
## Example: Programming questions

<table>
<thead>
<tr>
<th>information type</th>
<th>search times</th>
<th>% agreed info is...</th>
<th>frequency and outcome of searches</th>
<th>frequency of sources</th>
</tr>
</thead>
<tbody>
<tr>
<td>s1 Did I make any mistakes in my new code?</td>
<td>0 1 6</td>
<td>■ 59</td>
<td></td>
<td>br = bug report, dbg = debugger</td>
</tr>
<tr>
<td>a2 What have my coworkers been doing?</td>
<td>0 1 11</td>
<td>■ 17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>u3 What code caused this program state?</td>
<td>0 2 21</td>
<td>■ 90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r2 In what situations does this failure occur?</td>
<td>0 2 49</td>
<td>■ 80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d2 What is the program supposed to do?</td>
<td>0 1 21</td>
<td>■ 93</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a1 How have resources I depend on changed?</td>
<td>0 2 17</td>
<td>■ 73</td>
<td></td>
<td></td>
</tr>
<tr>
<td>u1 What code could have caused this behavior?</td>
<td>0 2 14</td>
<td>■ 71</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c2 How do I use this data structure or function?</td>
<td>0 2 21</td>
<td>■ 61</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d3 Why was this code implemented this way?</td>
<td>0 2 6</td>
<td>■ 44</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b3 Is this problem worth fixing?</td>
<td>0 2 9</td>
<td>■ 85</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d4 What are the implications of this change?</td>
<td>1 5</td>
<td>■ 56</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d1 What is the purpose of this code?</td>
<td>1 7</td>
<td>■ 66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b2 Is this a legitimate problem?</td>
<td>0 2</td>
<td>■ 49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s2 Did I follow my team's conventions?</td>
<td>0 7 25</td>
<td>■ 41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>r1 What does the failure look like?</td>
<td>0 0 2</td>
<td>■ 88</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s3 Which changes are part of this submission?</td>
<td>0 2 3</td>
<td>■ 61</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c3 How can I coordinate this with this other code?</td>
<td>1 1</td>
<td>■ 75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>b2 How difficult will this problem be to fix?</td>
<td>2 4</td>
<td>■ 41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c1 What can be used to implement this behavior?</td>
<td>2 2</td>
<td>■ 61</td>
<td></td>
<td></td>
</tr>
<tr>
<td>a3 What information was relevant to my task?</td>
<td>1 1</td>
<td>■ 59</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>this is a serious problem for me</th>
<th>% agree</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Code Understanding</strong></td>
<td></td>
</tr>
<tr>
<td>Understanding the rationale behind a piece of code</td>
<td>66%</td>
</tr>
<tr>
<td>Understanding code that someone else wrote</td>
<td>56%</td>
</tr>
<tr>
<td>Understanding the history of a piece of code</td>
<td>51%</td>
</tr>
<tr>
<td>Understanding code that I wrote a while ago</td>
<td>17%</td>
</tr>
<tr>
<td><strong>Task Switching</strong></td>
<td></td>
</tr>
<tr>
<td>Having to switch tasks often because of requests from my teammates or manager</td>
<td>62%</td>
</tr>
<tr>
<td>Having to switch tasks because my current task gets blocked</td>
<td>50%</td>
</tr>
<tr>
<td><strong>Modularity</strong></td>
<td></td>
</tr>
<tr>
<td>Being aware of changes to code elsewhere that impact my code</td>
<td>61%</td>
</tr>
<tr>
<td>Understanding the impact of changes I make on code elsewhere</td>
<td>55%</td>
</tr>
<tr>
<td><strong>Links between Artifacts</strong></td>
<td></td>
</tr>
<tr>
<td>Finding all the places code has been duplicated</td>
<td>59%</td>
</tr>
<tr>
<td>Understanding who “owns” a piece of code</td>
<td>50%</td>
</tr>
</tbody>
</table>

Questions developers report as hard to answer span many topics

Rationale (42)
Why was it done this way? [14] [15] [7]
Why wasn’t it done this other way? [15]  
Was this intentional, accidental, or a hack? [9] [15]  
How did this ever work? [4]  

Debugging (26)
How did this runtime state occur? [12] [15]  
What runtime state changed when this executed? [2]  
Where was this variable last changed? [1]  
How is this object different from that object? [1]  
Why didn’t this happen? [3]  
How do I debug this bug in this environment? [3]  
In what circumstances does this bug occur? [3] [15]  
Which team’s component caused this bug? [1]  

Intent and Implementation (32)
What is the intent of this code? [12] [15]  
What does this do (6) in this case (10)? [16] [24]  
How does it implement this behavior? [4] [24]  

Refactoring (25)
Is there functionality or code that could be refactored? [4]  
Is the existing design a good design? [2]  
Is it possible to refactor this? [9]  
How can I refactor this (2) without breaking existing users? [7] [9]  
Should I refactor this? [2]  
Are the benefits of this refactoring worth the time investment? [3]  

History (23)
When, how, by whom, and why was this code changed or inserted? [13] [7]  
What else changed when this code was changed or inserted? [2]  
How has it changed over time? [4] [7]  
Has this code always been this way? [2]  
What recent changes have been made? [1] [15] [7]  
Have changes in another branch been integrated into this branch? [1]  

Implications (21)
What are the implications of this change for (5) API clients (5), security (3), concurrency (3), performance (2), platforms (1), tests (1), or obfuscation (1)? [21] [13] [24]  

Testing (20)
Is this code correct? [6] [15]  
How can I test this code or functionality? [9]  
Is this tested? [3]  
Is the test or code responsible for this test failure? [1]  
Is the documentation wrong, or is the code wrong? [1]  

Implementing (19)
How do I implement this (8), given this constraint (2)? [10]  
Which function or object should I pick? [2]  
What’s the best design for implementing this? [7]  

Control flow (19)
In what situations or user scenarios is this called? [5] [15] [24]  
What parameter values does each situation pass to this method? [1]  
What parameter values could lead to this case? [1]  
Is this method or code path called by dynamic dispatch here? [6]  
How do calls flow across process boundaries? [1]  
How many recursive calls happen during this operation? [1]  
Is this method or code path called frequently, or is it dead? [4]  
What throws this exception? [1]  
What is catching this exception? [1]  

Contracts (17)
What assumptions about preconditions does this code make? [5]  
What assumptions about post(3)/pre(2)conditions can be made? [5]  
What exceptions or errors can this method generate? [2]  
What are the constraints or normal values of this variable? [2]  
What is the correct order for calling these methods or initializing these objects? [2]  
What is responsible for updating this field? [1]  

Performance (16)
What is the performance of this code (5) on a large, real dataset (3)? [8]  
Which part of this code takes the most time? [4]  
Can this method have high stack consumption from recursion? [1]  
How big is this in memory? [2]  
How many of these objects get created? [1]  

Teammates (16)
Who is the owner or expert for this code? [3] [7]  
How do I convince my teammates to do this the “right way”? [12]  
Did my teammates do this? [1]  

Policies (15)
What is the policy for doing this? [10] [24]  
Is this the correct policy for doing this? [2] [15]  
How is the allocation lifetime of this object maintained? [3]  

Type relationships (15)
What are the composition, ownership, or usage relationships of this type? [5] [24]  
What is this type’s type hierarchy? [4] [24]  
What implements this interface? [4] [24]  
Where is this method overridden? [2]  

Data flow (14)
What is the original source of this data? [2] [15]  
What code directly or indirectly uses this data? [5]  
Where is the data referenced by this variable modified? [2]  
Where can this global variable be changed? [1]  
Where is this data structure used (1) for this purpose? [1] [2] [24]  
What parts of this data structure are modified by this code? [1] [24]  
What resources is this code using? [1]  

Location (13)
Where is this functionality implemented? [5] [24]  
Is this functionality already implemented? [5] [15]  
Where is this defined? [3]  

Building and branching (11)
Should I branch or code against the main branch? [1]  
How can I move this code to this branch? [1]  
What do I need to include to build this? [3]  
What includes are unnecessary? [2]  
How do I build this without doing a full build? [1]  
Why did the build break? [2] [59]  
Which preprocessor definitions were active when this was built? [1]  

Architecture (11)
How does this code interact with libraries? [4]  
What is the architecture of the code base? [3]  
How is this functionality organized into layers? [1]  
Is our API understandable and flexible? [3]  

Concurrency (9)
What threads reach this code (4) or data structure (2)? [6]  
Is this class or method thread-safe? [2]  
What members of this class does this lock protect? [1]  

Dependencies (5)
What depends on this code or design decision? [4] [7]  
What does this code depend on? [1]  

Method properties (2)
How big is this code? [1]  
How overloaded are the parameters to this function? [1]
Many of these already have tools that support them

• Debugging

• Refactoring

• Design Rationale

• So if there's already a tool designed to support this, why is it still so hard??
Supporting information needs

• Debugging is hard.
  • Tool x claims to make debugging easier!

• Does tool x help?

• Depends…
  • Does tool x apply in the situations that make debugging challenging?
  • Do developers have the context they need to invoke tool x
  • Does tool x reliably produce the information required
  • Are the interactions for using tool x usable
Debugging (26)

- How did this runtime state occur? (12)
  - data, memory corruption, race conditions, hangs, crashes, failed API calls, test failures, null pointers

- Where was this variable last changed? (1)
- Why didn’t this happen? (3)

* omniscient debuggers

Record execution history
Provide interactions for browsing or searching

WhyLine [1]

directly supports all 3 questions in some situations

Debugging (26)

How do I debug this bug in this environment? (3)

In what circumstances does this bug occur? (3)

statistical debugging [1]

- Sample execution traces on user computers
- Find correlations between crashes and predicates

No need to reproduce environment on developer computer

Examine correlations between crashes and predicates

Debugging (26)

How is this object **different** from that object? (1)

What runtime state **changed** when this executed? (2)

Which **team’s** component caused this bug? (1)
Which team should I assign this bug to?
Rationale (42)

How did this **ever** work? (4)

**Why wasn’t it done this other way?** (15)

**Why** was it done this way? (14)

Was this **intentional**, accidental, or a **hack**? (9)

naming, code structure, inheritance relationships, where resources freed, code duplication, algorithm choice, optimization, parameter validation visibility, exception policies
Refactoring (25)

* Is the existing design a good design? (2)

smell detectors [1], metrics

Look for bad design idioms
Suggests developer refactor

- data clumps
- feature envy
- refused bequest
- typecast

- instanceof
- magic number
- long method
- large class

* Is there functionality or code that could be refactored? (4)

clone detectors [2]

Detects syntactically similar code
Suggests developer refactor

ComponentUI mui = new MultiButtonUI();
return MultiLookAndFeel.createUIs(mui, (MultiButtonUI) mui);

ComponentUI mui = new MultiColorChooserUI();
return MultiLookAndFeel.createUIs(mui, (MultiColorChooserUI) mui);

obsolete code, duplicated functionality, redundant data between equally accessible data structures


Should I refactor this? (1)

Are the benefits of this refactoring worth the time investment? (3)
Is it possible to refactor this? (9)

How can I refactor this (2) without breaking existing users? (7)

IDE refactoring automation
rename
move
pull up
push down
encapsulate field
convert local variable to field
....

changing a method’s scope, moving functionality between layers, changing semantics of config values, making operations more data driven, generalizing code to be more reusable

higher-level refactorings
Find a new strategy that makes question easier to answer
Example of a programming strategy

```
# This Strategy helps you merge 2 branches in github and resolve conflicts

# Required Tools and Environments
Installing git
Github account

Ongoing project which is progressing in at least 2 branches
Git repository that is not associated with Github

# Required Knowledge
Basic git command knowledge
Knowledge of how to work with terminal and run commands

STRATEGY GitMerge()
  # Open the terminal, and use cd(change directory) command to move to the local git project directory
  Open the terminal and navigate to your git project directory
  IF you are not in the master branch
    # Run the command git checkout master
c    checkout to the master branch
  IF you are in the master branch
    # To merge the second branch with the master branch run the command
    "git merge secondBranch", which secondBranch is the name of your git
    second branch
    Merge the two branches
  IF the merge has a conflict
    SET 'conflictedFiles' TO the project files that have a conflict
    FOR EACH 'file' IN 'conflictedFiles'
      DO fixConflict('file')
  # Run GIT STATUS to see the latest changes
  # Run GIT ADD
  # Run GIT COMMIT -m ""
  # Run GIT PUSH
  Commit and push the changes

RETURN nothing
```
Developers often have choices between strategies

**Question** Can I remove this call?

**Strategy: Implement & test**
Remove the call & test behavior change

**Strategy: Understand**
Understand implications before editing by investigating callees
Guess and check debugging

1. Describe in words how the program is failing

2. Brainstorm a list of possible causes of this failure

3. For each possible cause:
   
   1. Read the potentially defective code.
   
   2. Gather data about program execution to verify that it is the defect.
   
   3. If it is the defect, repair it.

4. If you didn’t find the defect, return to 2
STRATEGY debug()
1. Is the faulty output you're investigating printed to a command line?
2. If the faulty output is logged to a command line
3. To find print statements, try searching for keywords related to 'log' or 'print'
4. Set outputLines to the line numbers of calls to console logging functions
5. Graphical output includes things like colored lines and rectangles
6. If the faulty output is graphical output
7. To find these lines, try searching for keywords related to graphical output, like 'draw' or 'fill'. Focus on lines that directly render something, not on higher-level functions that indirectly call rendering functions.
8. Set outputLines to the line numbers of function calls that directly render graphics to the screen
9. Now that you have some lines that could have directly produced the faulty output, you're going to check each line, see if it executed, and then find the cause of it executing. If you're lucky, you only have one output line to check.
10. For each 'line' in 'outputLines'
11. If the program executed 'line'
12. Analyze the line to determine its role in the overall behavior of the program
13. Check for errors such as the wrong function being called, the wrong argument being passed to a function, the wrong variable being referenced, or a wrong operator being used.
14. If any part of 'line' is inconsistent with its purpose
15. You found the bug.
16. Return 'line'
17. If the output statement is not wrong, perhaps the line was not supposed to execute at all?
18. If 'line' was not supposed to execute at all
19. The conditional might be in the same function as the output statement, or it might have been a conditional in a function that called this function. Check the call stack if necessary by setting a breakpoint. Find the conditional that led this line to being executed.
20. Some value in the conditional's boolean expression must have been wrong. Which value was it?
21. Set 'wrongValue' to the value in the conditional's boolean expression that ultimately allowed the faulty output to execute.
22. We'll use another strategy to find the source of the incorrect value.
23. Return localizeWrongValue('wrongValue')
24. If the line was supposed to execute, but it executed with an incorrect value, find that value.
25. If 'line' executed with an incorrect value
26. Set 'wrongValue' to the incorrect value
27. We'll use another strategy to find the cause of the incorrect value.
28. Return localizeWrongValue('wrongValue')
29. If you made it to this line, then you must have missed something. Is it possible you made a mistake above? If so, go back and verify your work, because something caused the faulty output.
30. Return nothing
Many factors influence the effectiveness of a strategy in a situation.

<table>
<thead>
<tr>
<th>Influencing factor</th>
<th>Strategy: <em>Implement &amp; test</em></th>
<th>vs.</th>
<th>Strategy: <em>Understand</em></th>
</tr>
</thead>
<tbody>
<tr>
<td>Work style [Clarke+04]</td>
<td>Opportunistic</td>
<td></td>
<td>Systematic</td>
</tr>
<tr>
<td>Development process</td>
<td>Test-driven development</td>
<td></td>
<td>Few unit tests</td>
</tr>
<tr>
<td>Cost of bugs</td>
<td>Low</td>
<td></td>
<td>High</td>
</tr>
<tr>
<td>Time to implement</td>
<td>Easy to implement</td>
<td></td>
<td>Hard to implement</td>
</tr>
<tr>
<td>Difficulty of testing</td>
<td>An easily tested property (e.g., performance)</td>
<td></td>
<td>Non-functional property (e.g., testing usability)</td>
</tr>
<tr>
<td>Test execution time</td>
<td>Short-running test suites</td>
<td></td>
<td>Long-running test suites</td>
</tr>
</tbody>
</table>
Developers often rapidly switch between alternative actions or strategies

<table>
<thead>
<tr>
<th>Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method (m)</td>
<td>Where is method (m) generating an error?</td>
</tr>
<tr>
<td></td>
<td>Rapidly found method (m) implementing command</td>
</tr>
<tr>
<td></td>
<td>Unsure where it generated error</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static call traversal</td>
<td>Statically traversed calls looking for something that would generate error</td>
</tr>
<tr>
<td>Debugger</td>
<td>Tried debugger</td>
</tr>
<tr>
<td>Grep</td>
<td>Did string search for error, found it, but many callers</td>
</tr>
<tr>
<td>Debugger</td>
<td>Stepped in debugger to find something relevant</td>
</tr>
<tr>
<td>Static call traversal</td>
<td>Statically traversed calls to explore</td>
</tr>
<tr>
<td>Debugger</td>
<td>Went back to stepping debugger to inspect values</td>
</tr>
<tr>
<td></td>
<td>Found the answer</td>
</tr>
</tbody>
</table>

(66 minutes)
Developers often rapidly switch between alternative actions or strategies

Lacks **knowledge** to determine how these lines influence program behavior

Tries to recover **rationale**, but no explanation in check-in message

Tests might have identified a bug, but don't prove **absence**.

Teammates **remembered** another scenario.

**Strategy 1. Guess the answer.**
— *This was a quick hack, not a reasoned changed because otherwise they would have been removed. But what would break if they were here?*

**Strategy 2. Check code history.**
— *I commented these out 2 years ago along with many other changes. But why?*

**Strategy 3. Implement & test.**
— *Removed comments, all tests still pass. But did I break anything?*

**Strategy 4. Ask my teammates.**
— *Sent an email. Teammates replied with a description of a rare input which causes it to break. Success!*
Some strategies are more effective than others in a specific situation
Strategies can make a large difference in task performance
Teaching strategies

• If some strategies are more effective, maybe we could just teach them?

  • Write down lots of expert strategies.

  • Give developers the written version of the strategy?

• Developers are habitual, solve problems as they always have

  • Read it once, try to follow it, return to prior strategy
Scaffolding novel strategy use

Teaching Strategies to Developers

• Recruited 28 participants with JS experience and varying levels of industry experience (mean 2.5 years)

• Asked participants to debug a defect and complete a code design task, compared natural strategies against strategy shared through Strategy Tracker

• Most participants with Strategy Tracker switched strategy

  • Design: Decompose or Template --> TDD

  • Debugging: Forwards search --> backwards search

• Enabled debugging participants to make more progress
What's next: Supporting information needs

- Many of the next lectures will focus on information needs in specific programming activities
  - Editing code
  - Debugging
  - Navigating code
  - Refactoring
  - Code reuse
- Some will focus on specific tool approaches
  - Visualization
  - Crowdsourcing
  - Program synthesis
- Next week: Programming as communication