

Exploring code

Thomas LaToza



05-899D: Human Aspects of Software
Development (HASD)

Spring, 2011



Topics related to exploring code

6. Finding and understanding code

6.1 **Exploring code**

6.2 Reverse engineering

6.3 Reading code (program comprehension), mental models of programs, effects of expertise

7.6 Navigating working sets - navigating between code you've already found

8.3 Debugging

Why do developers explore code?

Before proposing change

Understand how code works to plan a change

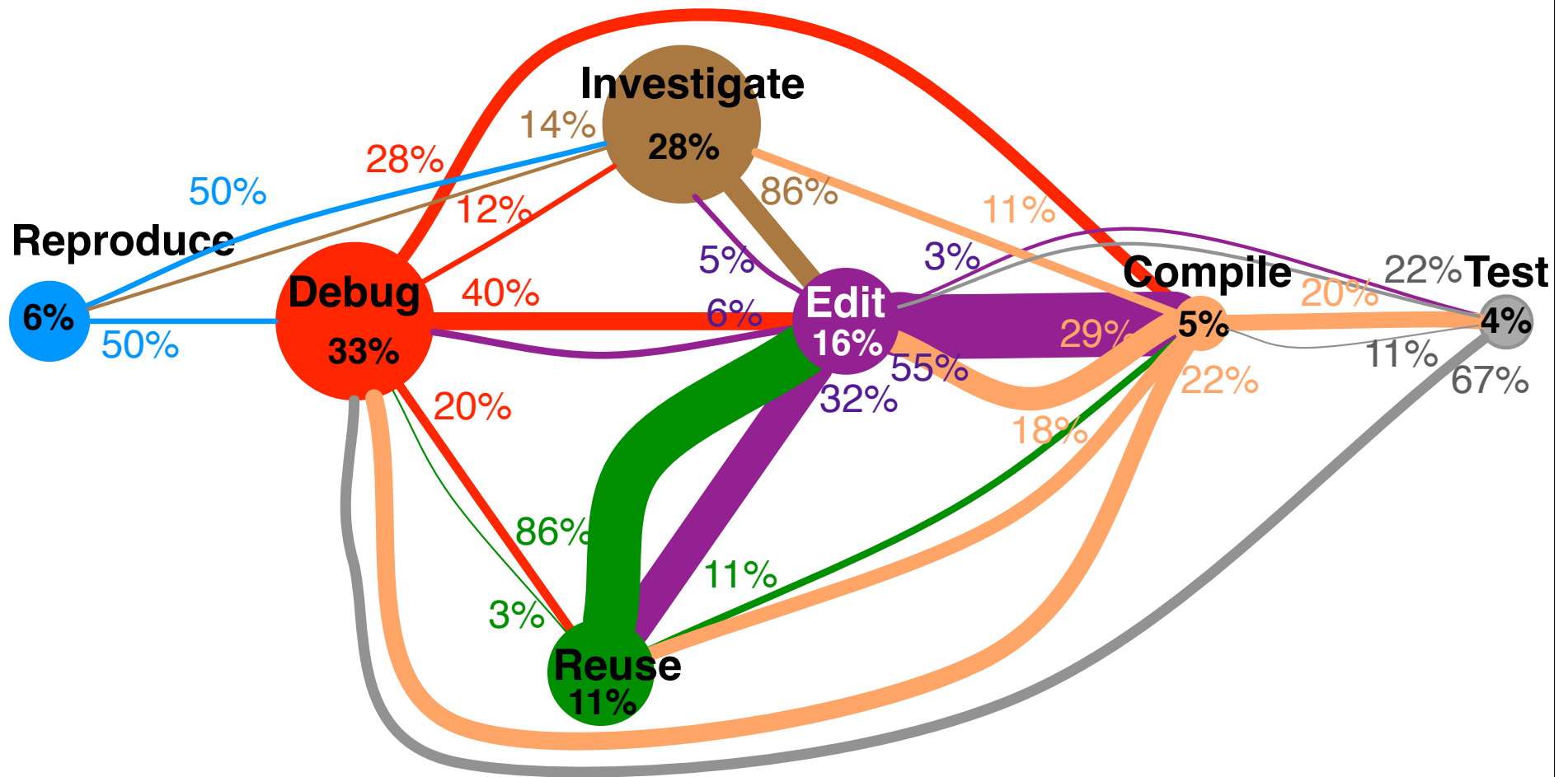
How should I do this? What's the right way to do this?

After proposing a change

Investigate the implications of a change

Will it work? What might it break?

“Investigating” code during coding activities



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

Exploratory studies related to exploring code

Research questions: Questions during coding activities, exploring code questions, problems exploring code

Study	Type	# developers	Task length (mins)	Tasks
[Vans, Mayrhauser, & Somlo 99]	Field observations	4	125	Fixing bugs
[LaToza, Garlan, Herbsleb, Myers 07]	Lab study	13	90 x 2	Investigate & fix complex design problems
[Ko, DeLine, & Venolia 07]	Field observations	17	90	Fixing bugs, implementing features
[Sillito, Murphy, & de Volder 08]	Lab study, field observations	9, 16	45, 30	Fixing bugs, implementing features
[Robillard, Coelho, & Murphy 08]	Lab study	5	125	Add autosave feature to text editor
[Starke, Luce, Sillito 09]	Lab study	10	30	Fixing bugs, implementing features
[Abi-Antoun, Ammar, & LaToza 10]	Lab study	3	120	Fixing bugs, implementing features
[LaToza & Myers 10]	Lab study, field observations, survey	13, 460, 17	90 mins x 2, 90 mins	Fixing bugs, implementing features
[LaToza & Myers 10]	Survey	179	N/A	N/A

Developers ask questions

information type	search times			% agreed info is...			frequency and outcome of searches				frequency of sources
	min	mid	max	import.	unavail.	inacc.	acquired	deferred	gave up	beyond obs.	br = bug report, dbug = debugger
s1 Did I make any mistakes in my new code?	0	1	6	■ 59	■ 7	■ 12				dbug 10 compile 26 intuition 6 unit test 4
a2 What have my coworkers been doing?	0	1	11	■ 17	■ 10	■ 10				coworker 20 email 13 tool 4 bug alert 4 im 2
u3 What code caused this program state?	0	2	21	■ 90	■ 49	■ 32				dbug 16 br 3 intuition 3 log 3 tools 3 code 2 coworker 1
r2 In what situations does this failure occur?	0	2	49	■ 80	■ 32	■ 20				br 8 coworker 8 inference 5 tools 3 dbug 2 comment 1
d2 What is the program supposed to do?	0	1	21	■ 93	■ 29	■ 29				spec 13 coworker 9 docs 5 email 1
a1 How have resources I depend on changed?	0	1	9	■ 41	■ 15	■ 15				tools 12 coworker 6 email 4 br 2 code 1
u1 What code could have caused this behavior?	0	2	17	■ 73	■ 20	■ 22				coworker 5 intuition 4 log 4 br 4 dbug 2 im 1 code 1 spec 1
c2 How do I use this data structure or function?	0	1	14	■ 71	■ 20	■ 29				docs 11 code 5 coworker 4 spec 1
d3 Why was this code implemented this way?	0	2	21	■ 61	■ 37	■ 39				code 4 intuition 4 history 3 coworker 2 dbug 2 tools 2 comment 1 br 1
b3 Is this problem worth fixing?	0	2	6	■ 44	■ 10	■ 20				coworker 12 email 2 br 1 intuition 1
d4 What are the implications of this change?	0	2	9	■ 85	■ 44	■ 49				coworker 13 log 1
d1 What is the purpose of this code?	1	1	5	■ 56	■ 24	■ 29				intuition 5 code 2 dbug 2 tools 2 spec 1 docs 1
u2 What's statically related to this code?	0	1	7	■ 66	■ 27	■ 27				tools 8 intuition 2 email 1
b1 Is this a legitimate problem?	0	1	2	■ 49	■ 17	■ 34				br 5 coworker 1 log 1
s2 Did I follow my team's conventions?	0	7	25	■ 41	■ 10	■ 15				docs 2 tools 2 memory 1
r1 What does the failure look like?	0	0	2	■ 88	■ 24	■ 23				br 3 screenshot 2
s3 Which changes are part of this submission?	0	2	3	■ 61	■ 7	■ 5				tools 2 memory 2
c3 How I can coordinate this with this other code?	1	1	4	■ 75	■ 28	■ 30				docs 2 code 1 coworker 1
b2 How difficult will this problem be to fix?	2	2	4	■ 41	■ 15	■ 32				code 1 coworker 1 screenshot 1
c1 What can be used to implement this behavior?	2	2	2	■ 61	■ 27	■ 22				memory 1 docs 1
a3 What information was relevant to my task?	1	1	1	■ 59	■ 15	■ 13				memory 2

Andrew J. Ko, Robert DeLine, and Gina Venolia, [Information Needs in Collocated Software Development Teams](#), in ICSE '07: Proceedings of the 29th international conference on Software Engineering, IEEE Computer Society, Washington, DC, USA, May 2007.

Questions about code elements

Finding focus points

- 1 Which type represents this domain concept or this UI element or action?**
Full support: Lexical or static analysis based search (e.g., Eclipse's Open Type tool)
 - 2 Where in the code is the text in this error message or UI element?**
Full support: Lexical or static analysis based search (e.g., grep)
 - 3 Where is there any code involved in the implementation of this behavior?**
Partial support: Search with debugging support or feature location techniques (e.g., RECON3 [25])
 - 4 Is there a precedent or exemplar for this?**
Partial support: Lexical or static analysis based search or example finding tools (e.g., CodeFinder, assuming an appropriate repository this could be considered *full* support [22])
 - 5 Is there an entity named something like this in that unit (project, package or class, say)?**
Full support: Lexical or static analysis based search (e.g., grep)
-

Expanding focus points

- 6 What are the parts of this type?**
Partial support: Source code editors or structural overviews (e.g., Eclipse's Outline View)
 - 7 Which types is this type a part of?**
Full support: Cross-referencing tools (e.g., Masterscope [65])
 - 8 Where does this type fit in the type hierarchy?**
Full support: Type hierarchy tools (e.g., Eclipse's Type Hierarchy)
 - 9 Does this type have any siblings in the type hierarchy?**
Full support: Static analysis based type hierarchy tools (e.g., Eclipse's Type Hierarchy)
 - 10 Where is this field declared in the type hierarchy?**
Full support: Static analysis based type hierarchy tools (e.g., Eclipse's Type Hierarchy)
 - 11 Who implements this interface or these abstract methods?**
Full support: Static analysis based type hierarchy tools (e.g., Eclipse's search for implementors feature)
 - 12 Where is this method called or type referenced?**
Full support: Cross-referencing tools (e.g., CScope [58])
 - 13 When during the execution is this method called?**
Full support: Debugging tools (e.g., GDB)
 - 14 Where are instances of this class created?**
Full support: Cross-referencing tools (e.g., CScope [58])
 - 15 Where is this variable or data structure being accessed?**
Full support: Cross-referencing tools or slicing techniques (e.g., [20])
 - 16 What data can we access from this object?**
Partial support: Source code editors or structural overviews (e.g., Eclipse's Outline View)
 - 17 What does the declaration or definition of this look like?**
Full support: Lexical based cross-referencing tools (e.g., ctags)
 - 18 What are the arguments to this function?**
Full support: Lexical based cross-referencing tools (e.g., ctags)
 - 19 What are the values of these arguments at runtime?**
Full support: Debugging tools (e.g., GDB)
 - 20 What data is being modified in this code?**
Partial support: Source code editor, data-flow analysis techniques or slicing techniques (e.g., [20])
-

Jonathan Sillito, Gail C. Murphy and Kris De Volder. [Asking and Answering Questions during a Programming Change Task](#). In IEEE Transactions on Software Engineering. 2008.

Questions about code elements

Understanding a subgraph

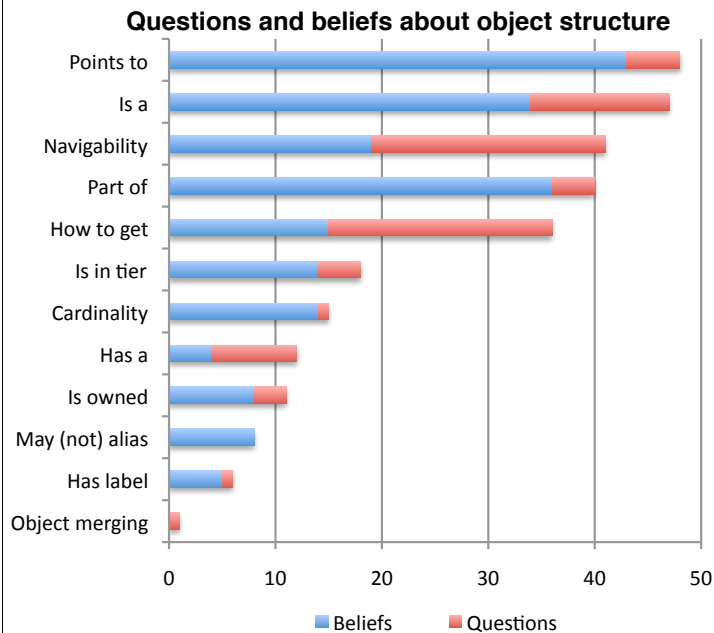
- 21 **How are instances of these types created and assembled?**
Partial support: Visualization or browsing tools (e.g., SHriMP [61])
- 22 **How are these types or objects related? (whole-part)**
Partial support: Visualization or browsing tools (e.g., SHriMP [61])
- 23 **How is this feature or concern (object ownership, UI control, etc.) implemented?**
Partial support: Feature location techniques (e.g., software reconnaissance [72])
- 24 **What in this structure distinguishes these cases?**
Partial support: Visualization or browsing tools (e.g., FEAT [49])
- 25 **What is the behavior that these types provide together and how is it distributed over the types?**
Partial support: Visualization or browsing tools (e.g., Relo [55])
- 26 **What is the “correct” way to use or access this data structure?**
Partial support: Visualization or browsing tools (e.g., SHriMP [61])
- 27 **How does this data structure look at runtime?**
Partial support: Debugging and data structure visualization tools (e.g., Amethyst [43])
- 28 **How can data be passed to (or accessed at) this point in the code?**
Partial support: Runtime visualization tools (e.g., BALSAs [6])
- 29 **How is control getting (from here to) here?**
Partial support: Visualization or browsing tools (e.g., a call hierarchy browser)
- 30 **Why isn’t control reaching this point in the code?**
Partial support: Debugging and slicing techniques (e.g., Whyline [32])
- 31 **Which execution path is being taken in this case?**
Partial support: Debugging and slicing techniques (e.g., GDB)
- 32 **Under what circumstances is this method called or exception thrown?**
Partial support: Debugging and visualization tools (e.g., BALSAs [6])
- 33 **What parts of this data structure are accessed in this code?**
Partial support: Source code editors or browsing (e.g., Eclipse’s Outline View)

Questions over groups of subgraphs

- 34 **How does the system behavior vary over these types or cases?**
Partial support: Structural searches and debugging techniques (e.g., GDB)
 - 35 **What are the differences between these files or types?**
Partial support: Line-based comparison tools (e.g., diff)
 - 36 **What is the difference between these similar parts of the code (e.g., between sets of methods)?**
Partial support: Line-based comparison tools (e.g., diff)
 - 37 **What is the mapping between these UI types and these model types?**
Partial support: Conceptual module querying [2]
 - 38 **Where should this branch be inserted or how should this case be handled?**
Partial support: Visualization and browsing tools (e.g., SHriMP [61])
 - 39 **Where in the UI should this functionality be added?**
Partial support: Visualization and browsing tools (e.g., SHriMP [61])
 - 40 **To move this feature into this code what else needs to be moved?**
Partial support: Conceptual module querying [2]
 - 41 **How can we know that this object has been created and initialized correctly?**
Partial support: Visualization and browsing tools (e.g., SHriMP [61])
 - 42 **What will be (or has been) the direct impact of this change?**
Partial support: Testing and impact analysis techniques (e.g., Chianti [47])
 - 43 **What will be the total impact of this change?**
Partial support: Testing and impact analysis techniques (e.g., Unit tests)
 - 44 **Will this completely solve the problem or provide the enhancement?**
Partial support: Testing techniques [3]
-

Jonathan Sillito, Gail C. Murphy and Kris De Volder. [Asking and Answering Questions during a Programming Change Task](#). In IEEE Transactions on Software Engineering. 2008.

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. I know I need to get the view from here; so how do I do that?
Is in tier	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. 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
Is owned	[...] 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.
May alias	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...

Abi-Antoun, M., Ammar, N., LaToza, T. (2010). [Questions about object structure during coding activities](#). In the Workshop on Cooperative and Human Aspects of Software Engineering at ICSE '10.

Hard-to-answer questions about code

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? (1)
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) [15][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? (3) [15][24]
What parameter values does each situation pass to this method? (1)
What parameter values could lead to this case? (1)
What are the possible actual methods 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 pre(3)/post(2)conditions can be made?
What exceptions or errors can this method generate? (2)
What are the constraints on 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)*

Hard-to-answer questions about code

No **dominant** hard question

Rationale questions both most frequently reported and unaddressed by research

Many hard-to-answer questions already **addressed** by research

Many hard-to-answer questions **are not** addressed by research

Gaps between **situations** addressed by research and problems developers reported

Questions about control flow (19)

✘ In what **situations** or user scenarios is this called? (3)

✘ How do **calls** flow across process boundaries? (1)

* Is this method or code path called frequently, or is it **dead**? (4)
profiling, program analysis

✘ What parameter values does each **situation** pass to this method? (1)

✘ What parameter values could **lead** to this case? (1)

* How many **recursive** calls happen during this operation? (1)
profilers

* What **throws** this exception? (1)

* What is **catching** this exception? (1)
checked exceptions

* What are the possible actual methods called by **dynamic dispatch** here? (6)
IDE code browsers

Developers subgoal

Fix bug

Repro bug - “500 internal server error”

Debug using VS

Attach debugger

Set to break on exceptions

NPE exception - why is this null?

Browse callers using emacs & ctags

Edit values in VS and try continuing

Edit & test - still same exception

Browse further up dataflow path - find mutexes

Don't own code, so not sure how supposed to work

Talk to teammate who owns code

Make a change

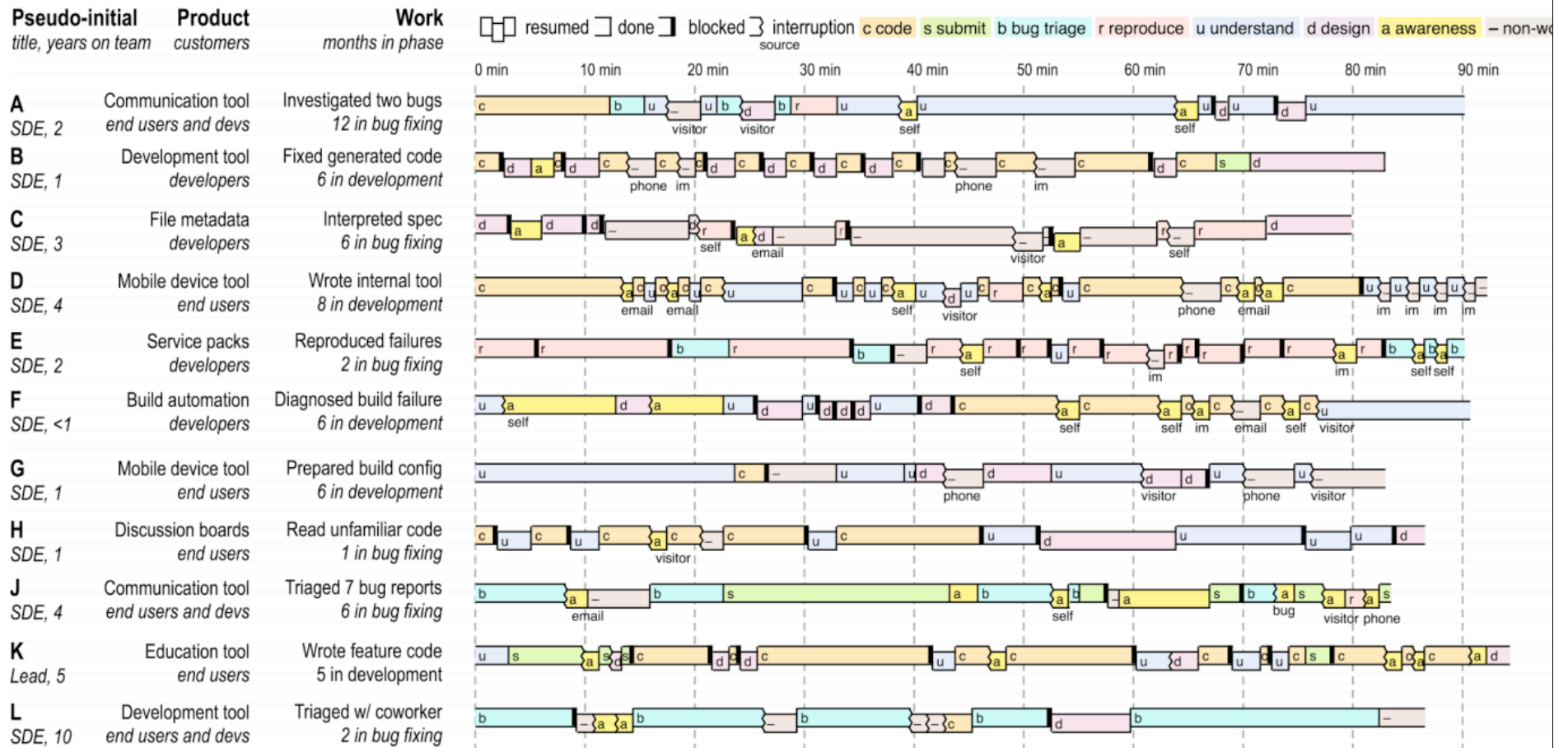
Goals not strictly hierarchic - actions may serve multiple functions

Thomas D. LaToza, Gina Venolia, and Robert DeLine. 2006. Maintaining mental models: a study of developer work habits. In *Proceedings of the 28th international conference on Software engineering (ICSE '06)*. ACM, New York, NY, USA, 492-501.

Questions evolve through a task

- 1 Our participants approached a corrective maintenance task by first forming hypotheses about what the problem is based on their past experience. These initial hypotheses were often correct and guided much of their work on the task.
- 2 From their hypothesis, our participants formed search queries based on experience and expectations around naming conventions. Sometimes trying multiple searches based on synonyms or various ways to express similar ideas.
- 3 Our participants often had only a fuzzy notion of what to search for (or what to look for in a result set) especially early in a task. Often their searches were very general and returned many results.
- 4 Rather than systematically investigate search results, our participants generally skimmed through results looking for evidence of relevance (based on their hypotheses), at times focusing on a package that was consistent with their guiding hypothesis.
- 5 Our participants opened a small number of results in the source code editor and rarely return to their search results (i.e., not iterating between results and elements). Most often the opened element was only skimmed.

Developers constantly switch tasks



Task switches occurred on average every **5** minutes (+/- 1.7)

Caused by emails, bug database alerts, waiting for compilation or tests to run, email replies, needs to learn something,

Andrew J. Ko, Robert DeLine, and Gina Venolia, [Information Needs in Collocated Software Development Teams](#), in ICSE '07: Proceedings of the 29th international conference on Software Engineering, IEEE Computer Society, Washington, DC, USA, May 2007.

High-level vs. low level questions

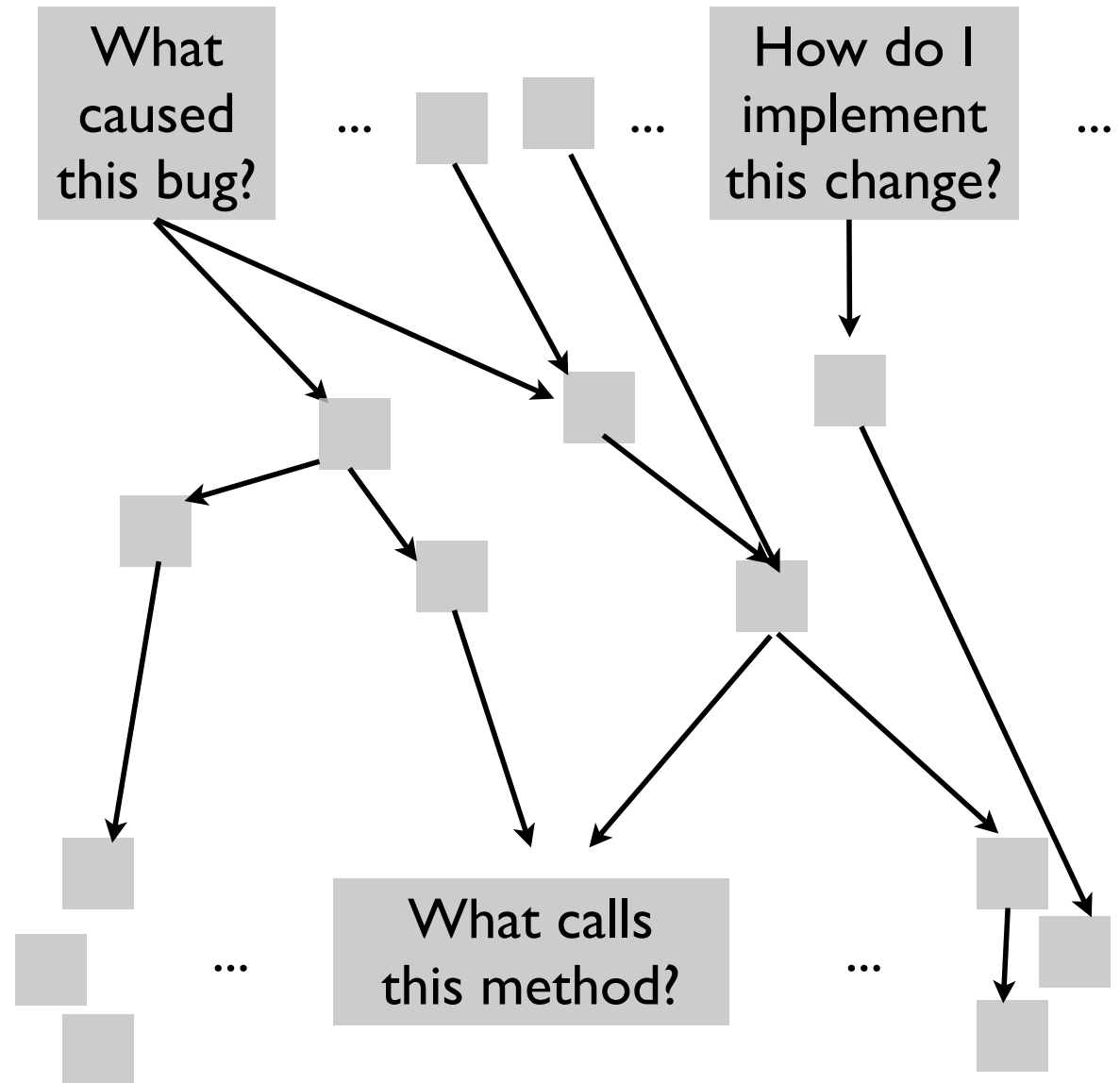
High level questions

harder to answer
conceptual
hard to support

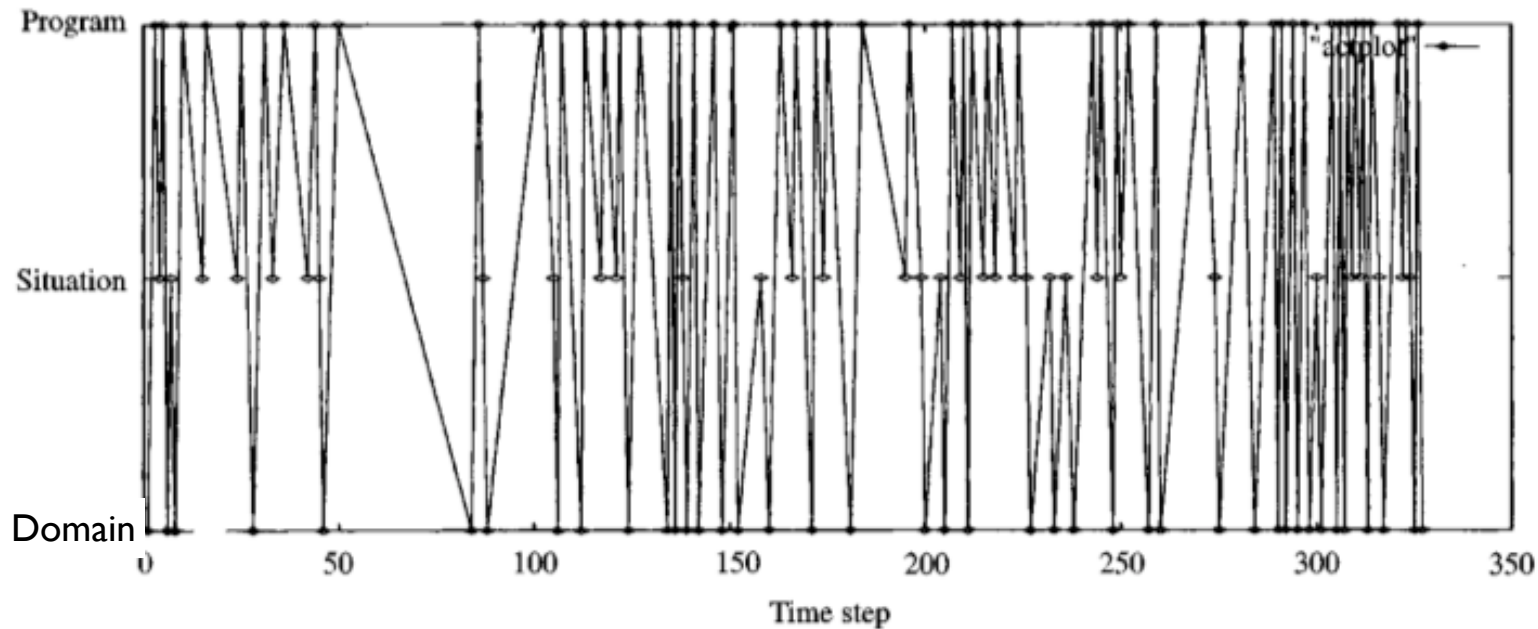
Mid level questions

Low level questions

easier to answer
behavioral
direct tool support



Developers switch between mental models of code



Domain model - knowledge about how application domain works (e.g., how distributed systems work)

Situation model - contracts of what code should do, functional model of how these are connected

Program model - control flow, syntax structures, implementation details

A. Marie Vans, Anneliese von Mayhauser, and Gabriel Somlo. 1999. Program understanding behavior during corrective maintenance of large-scale software. *Int. J. Hum.-Comput. Stud.* 51, 1 (July 1999), 31-70.

Developers explain design decisions

“When you're inserting text you could actually doing something that makes the folds status wrong. ... In the quick brown fox. If fox is under brown and I'm right at fox and I hit backspace. Then I would need to update my fold display to reflect the new reality which is that it's in a different place. It's now a child of quick, not a child of brown.”

A scenario

Developer considering a change.

But is current design a) designed to satisfy some (unknown) constraint or b) accidental?

Explanations establish **traceability** from low level design decisions to motivating requirements.

Thomas D. LaToza, David Garlan, James D. Herbsleb, and Brad A. Myers. 2007. [Program comprehension as fact finding](#). In ESEC/FSE 07, 361-370.

Exploring code involves guesswork

Developers **hypothesize** facts

Sometime look for evidence to support or reject hypothesis

How much is enough?

Sometimes just make an assumption

If decision might be **constrained**, developer could

- a) keep investigating potential constraint
- b) assume true, make change
- c) assume false, make change

Developers try to **predict** success.

“Why wouldn’t they call it? Now, can I test this? So why if you know the answer to the problem, do you put the code in the wrong place and then leave a comment? That’s not like these people.”

Developers choose strategies to answer questions

A **strategy** is a sequence of actions developers use to accomplish a goal. Developers answer **questions** by selecting a strategy, performing its actions, and asking further questions.

Influencing factor	Strategy: <i>Implement & test</i>	vs.	Strategy: <i>Understand</i>
Work style [Clarke+04]	Opportunistic		Systematic
Development process	Test-driven development		Few unit tests
Cost of bugs	Low		High
Time to implement	Easy to implement		Hard to implement
Difficulty of testing	An easily tested property (e.g., performance)		Non-functional property (e.g., testing usability)
Test execution time	Short-running test suites		Long-running test suites

LaToza, T. D., & Myers, B. A. (2010). On the importance of understanding the strategies that developers use. In the Workshop on Cooperative and Human Aspects of Software Engineering at ICSE '10.

Developers use multiple strategies to answer a single question

In order to answer a question, we observed a developer **switching** through four strategies.

Question *Why were these four lines commented out? Why can't I add them back?*

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

Strategy 1. *Guess the answer.*

— *This was a quick hack, not a reasoned change because otherwise they would have been removed. But what would break if they were here?*

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

Strategy 2. *Check code history.*

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

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

Strategy 3. *Implement & test.*

— *Removed comments, all tests still pass. But did I break anything?*

Teammates **remembered** another scenario.

Strategy 4. *Ask my teammates.*

— *Sent an email. Teammates replied with a description of a rare input which causes it to break. Success!*

LaToza, T. D., & Myers, B. A. (2010). On the importance of understanding the strategies that developers use. In the Workshop on Cooperative and Human Aspects of Software Engineering at ICSE '10.

How effective developers investigate code

1. Unsuccessful participants did not understand **where** the design implied changes should go and made changes in one place.
2. Unless they had a relevant question, **stumbling** into the code they should be looking at didn't help.
3. Successful developers created a detailed **plan**.
4. Successful developers reinvestigated methods less frequently.
5. Successful developers used more keyword & reference **searches** rather than browsing or scrolling through lists.

Limitations: based on 5 participants on one lab task. Listed as **hypotheses** rather than results.

Martin P. Robillard, Wesley Coelho, and Gail C. Murphy. [How Effective Developers Investigate Source Code: An Exploratory Study](#). IEEE Transactions on Software Engineering, 30(12):889-903, December 2004.

Experienced developers work more effectively

Better **predicted** which parts of the code might be relevant

Chunked code as “caching”

Expert after 1 min: “Well this is just updating a cache. So, what we’re upset about is that you want to issue an event and you are doing it by forcing an update of the cache for the fold level of a particular line.”

Novice professional after 10: “What it did was ... computes the new line number and fires an event. But I didn’t see it change any state.”

Explained facts novices could not

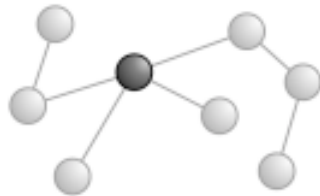
“What’s going on is that when you’re inserting text you could actually be doing something that makes the folds status wrong. So, if in our example here, in the quick brown fox. If fox is under brown and I’m right at fox and I hit backspace. Then I would need to update my fold display to reflect the new reality, which is that it’s in a different place.”

Figured out and made the same changes faster

a(0)	d(1.5)	g(2.5)	h(2.5)	j(3)	i(3)	K(3)	L(10)	
10	13	4	11	9	4	3	4	mins to extract getFoldLevel, sorted by years of experience (in parens)

Successfully solved the design problem rather than its symptoms

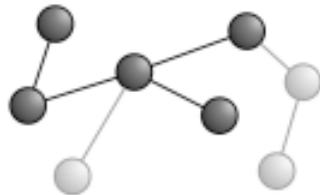
Developers find focus points and work outwards



Finding focus points

5 kinds of questions.

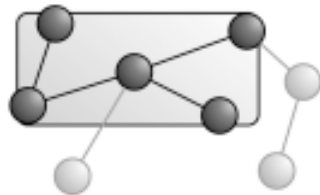
For example: Which type represents this domain concept?



Expanding focus points

15 kinds of questions.

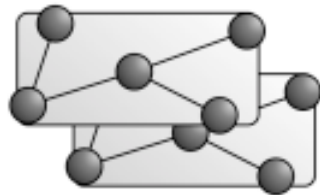
For example: Which types is this type a part of?



Understanding a subgraph

13 kinds of questions.

For example: What is the behavior these types provide together?



Questions over groups of subgraphs

11 kinds of questions.

For example: What is the mapping between these UI types and these model types?

Developers search along control flow

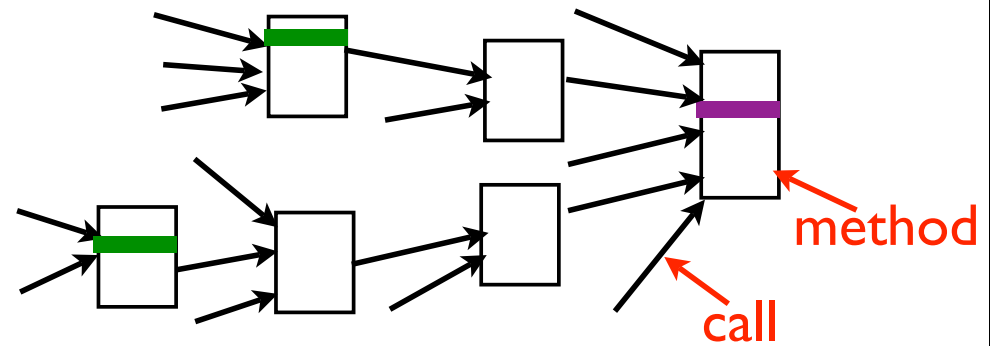
A developer asked:

What I'd like to do is identify those core...**events**... .
[But] I'm concerned that I won't get all of the **events**
that cause this...to get **updated**. And I'm not sure,
with the existing tools in Eclipse, how to find...the
places that can cause this **thing** to be called.

Question: What **events** trigger this **update**?

Reachability question:

Search upstream across
paths to **update** for **events**



50% of bugs were related to reachability questions

False assumption

Related reachability question

Method m is fast enough that it does not matter that it is called more frequently.

find ends in traces(jEdit, m_{start} , m_{end} , ?)

Method m need not invoke method n as it is only called in a situation in which n is already called. (**2 bugs**)

find callers(m) in traces(jEdit, ?, m , ?)

The scroll handler a does not need to notify b , because b is unrelated to scrolling.

find grep("scroll") in traces(jEdit, a_{start} , a_{end} , ?)

Removing this call in m does not influence behavior downstream.

*compare(traces(jEdit_{old}, m_{start} , ?, ?),
traces(jEdit_{new}, m_{start} , ?, ?))*

Question answered incorrectly

From what callers can the guards protecting statement d in method m be true?

find callers(m) in traces(jEdit, ?, d , ?)

Why is calling m necessary?

find ends in traces(jEdit, m_{start} , m_{end} , ?)

What situations currently trigger this computation in m ?

find ends in traces(jEdit, ?, m , ?)

4 out of 5 longest investigation activities

Primary question	Time (mins)	Related reachability question
How is this data structure being mutated in this code?	83	Search downstream for writes to data structure
“Where [is] the code assuming that the tables are already there?”	53	Compare behaviors when tables are or are not loaded
“How [does] application state change when <i>m</i> is called denoting startup completion?”	50	Find field writes caused by <i>m</i>
“Is [there] another reason why <i>status</i> could be non-zero?”	11	Find statements through which values flow into status

Conclusions: Exploring code is “messy”

Developers constantly change
tasks as interrupted
questions to answer evolve over time
strategy to answer question
mental model (domain, situation, program) as questions change

Skim through results rather than methodically read results

Deal with uncertain answers to questions

Many factors may influence which strategy developers use

Tools for supporting exploring code

Build a graph of nodes (methods, types, ...) connected by various **relationships** (calls, data dependence, references,)

Support **queries** and/or **traversing** relationships

Tools differ in
relationships supported
type of traversals supported

Evaluations in tool papers all case studies
Do these tools really **help**?

CodeSurfer: Traversing slices

The screenshot displays two windows from the CodeSurfer IDE. The left window, titled 'Project Find', shows search criteria: Variable: 'entry', Function: 'acl_getlimit', and File: (empty). It lists 13 matches in a table:

File	Line	Function	Kind	Item
src/acl.c	65	getaclentry	Vertex	*next != 0
src/acl.c	67	getaclentry	Vertex	*next != 0
src/acl.c	66	getaclentry	Vertex	*next = *next->next
src/access.c	775	acl_getlimit	Vertex	entry->arg[0] != 0
src/access.c	775	acl_getlimit	Vertex	entry->arg[1] != 0
src/access.c	775	acl_getlimit	Vertex	entry->arg[2] != 0

The right window, titled 'src/ftpd.c', shows the source code for 'timeout_data'. A context menu is open over the variable, listing options like 'Navigation', 'Properties', 'Call Graph', and 'Queries'. The 'Data Predecessors' and 'Data Successors' sections are expanded, showing 'timeout_data = 1200 [expression]' and 'timeout_data = value [expression]' respectively.

Anderson, P.; Reps, T.; Teitelbaum, T.; , "[Design and implementation of a fine-grained software inspection tool](#)," *Software Engineering, IEEE Transactions on* , vol.29, no.8, pp. 721- 733, Aug. 2003.

Querying multiple information sources

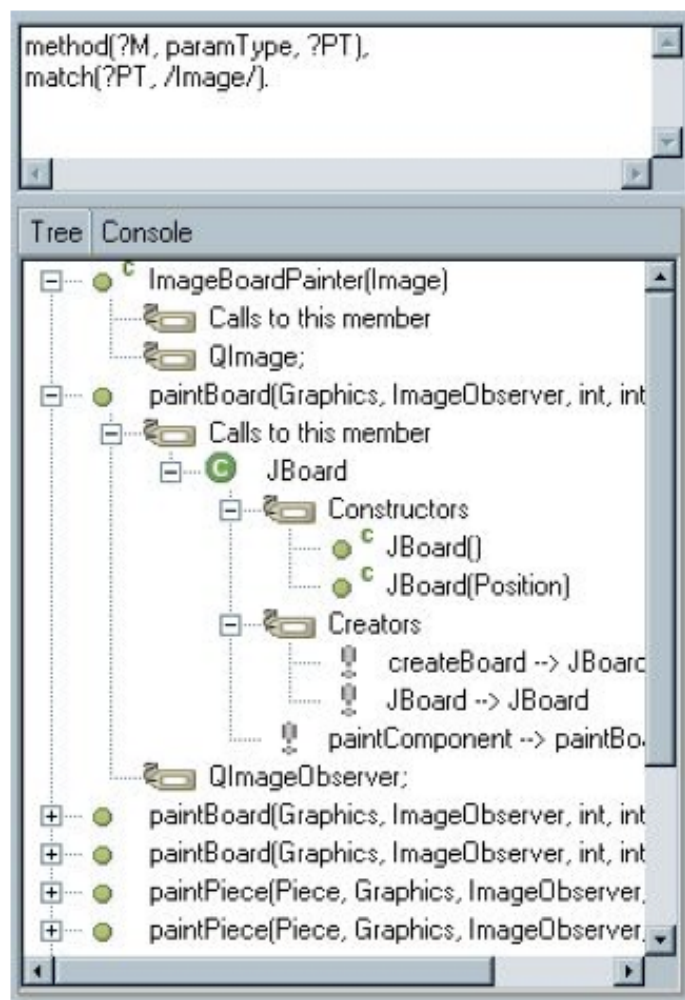
Category / Conceptual Queries	Contributing Spheres			Usage	
	Static	Dynamic	Evolution Plug-ins	#	%
<i>inter-class</i>					
1 †What methods return instances of this type? [16]	✓			1	0.4%
2 *What methods instantiate this type? [27]	✓	✓		8	2.9%
3 *What methods take an argument of this type? [27]	✓			2	0.7%
4 What fields are declared as being of this type?	✓			8	2.9%
5 *What calls this method? [27]	✓	✓		78	28.2%
6 *Where is the value of this field retrieved? [27]	✓			23	8.3%
7 *Where is the value of this field set? [27]	✓			5	1.8%
8 *What tests instanceof against this type? [17]	✓			2	0.7%
9 *Where are casts to this type? [17]	✓			1	0.4%
10 *What throws exceptions of this type? [27]	✓			0	0%
11 *What catches exceptions of this type?	✓			0	0%
12 *What references this type by name? [27]	✓			13	4.7%
13 *Which of the classes in this package were <i>actually</i> used?		✓		7	2.5%
14 *Which methods defined by this class were <i>actually</i> used?		✓		5	1.8%
<i>intra-class</i>					
15 ‡*What fields does this type or method access? [27]	✓			16	5.8%
16 ‡*What methods does this type or method call? [27]	✓	✓		23	8.3%
17 †*What types does this {type, method, or extension} reference?	✓		✓	18	6.5%
<i>inheritance</i>					
18 *What interfaces does this type implement? [27]	✓			0	0%
19 What are this class' subclasses? [27]	✓			6	2.2%
20 *What classes implement this interface? [27]	✓			8	2.9%
21 *What interfaces extend this interface? [27]	✓			5	1.8%
22 †What are this type's siblings? [27]	✓			2	0.7%
<i>declarations</i>					
23 What are all the fields are declared by this type? [27]	✓			9	3.2%
24 What class methods implement this interface method? [27]	✓			6	2.2%
25 *What interface methods specify this class method? [27]	✓			2	0.7%
26 What class methods does this method override?	✓			5	1.8%
27 What class methods override this method?	✓			7	2.5%
28 ‡What extension points or extensions reference this {type, file, folder}?			✓	6	2.2%
29 †What types is this type adaptable to?	✓		✓	3	1.1%
30 †What types could this type have be adapted from?	✓		✓	2	0.7%
31 What extensions are there of this extension point?			✓	4	1.4%
32 What plug-ins depend on this plug-in?			✓	2	0.7%
<i>evolution</i>					
33 †What transactions changed this element? [21]			✓	N/A	N/A
34 †Who has changed this element, and when? [21, 31]			✓	N/A	N/A
35 †What elements were changed in this transaction? [21]			✓	N/A	N/A
36 †What files were changed in this transaction?			✓	N/A	N/A

JQuery: Querying & Traversing Relationships

Hypothesized problems:

1. Exploration path lost as developers perform multiple searches, use different tools. Leads to disorientation.
2. Query tools support many types of relationships, but leads to query / analyze / new query cycle that loses exploration path.

JQuery: Querying & Traversing Relationships



Node Type	Relationships
Packages	Classes
	Interfaces
	All Types
Classes	Methods
	Fields
	Subtypes
	Supertypes
	Imports
	Constructors
	Creators
	References Methods
	References Types
	Calls to this type
Methods	References Methods
	References Types
	References Fields
	Calls to this method
	Signature
References	Caller
	Callee
	Caller's callers
	Callee's callees

Evaluation: Series of 3 examples

D. Janzen and K. D. Volder. [Navigating and querying code without getting lost](#). In Proc. Int'l Conf. Aspect-Oriented Softw. Development, pages 178–187. ACM, 2003.

Natural language querying

Problem: lots of IDE features,
novices may not know which one
to use to answer a question

Solution: natural language query
interface

Information in ontology

Class: Class

- hasMethod Method
- hasAttribute Attribute
- isReturnTypeOf Method
- isSubclassOf Class
- isSuperclassOf Class
- hasName String

Class: Method

- accessesAttribute Attribute
- hasParameter Parameter
- invokesMethod Method
- hasReturnType Class
- isInvokedByMethod Method
- isMethodOf Class
- hasName String

Class: Attribute

- isAttributeOf Class
- isAccessedByMethod Method
- hasName String

Class: Parameter

- isParameterOf Method
- hasName String

Ask a qu | Ask a question: | a question: | estion: | on: | | estion:

wh | what a | are t | the c | callers of | of addC | the caller of addChart ?

what
where
which
white ①

accesses
are
argument
arguments
attribute
attributes ②

the
type
types ③

callees
callers
classes ④

of ⑤

addCategoryLabelToolTip
addChangeListener
addChart
addChartMouseListener
addChoosableFileFilter
addCornerTextItem ⑥

? ⑦

org.jfree.chart.servlet - src - JFreeChart 1562

ServletUtilities 1562

registerChartForDeletion(File, HttpSession)

Michael Würsch, Giacomo Ghezzi, Gerald Reif, and Harald C. Gall. 2010. [Supporting developers with natural language queries](#). In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, 165-174.

These code exploration tools may not help

Evaluated JQuery, Ferret, Suade code exploration tools

18 professional developers working in the lab adding features

“No evidence of **any** practical effect”

Did not reduce the developer’s perceived mental **workload**

Did not make exploration more **focused** and reduce # of program elements viewed

Did not help developers find more of the salient locations

