Questions Programmers Ask During Software Evolution Tasks

Jonathan Sillito*, Gail C. Murphy and Kris De Volder
University of British Columbia

* now at the University of Calgary
Emacs ~1970s

GNU Emacs is one component of the GNU/Linux operating system. You can do basic editing with the menu bar and scroll bar using the mouse.

Important Help menu items:
Emacs Tutorial Learn how to use Emacs efficiently
Emacs FAQ Frequently asked questions and answers
Read the Emacs Manual View the Emacs manual using Info
(Non)Warranty GNU Emacs comes with ABSOLUTELY NO WARRANTY
Copying Conditions Conditions for redistributing and changing Emacs
Getting New Versions How to obtain the latest version of Emacs
More Manuals / Ordering Manuals Buying printed manuals from the FSF

This is GNU Emacs 22.0.50.1 (i686-pc-linux-gnu, GTK+ Version 2.8.17)
of 2006-05-08 on shogouki
Copyright 1989 Free Software Foundation, Inc.

if an Emacs session crashed recently, type M-x recover-session RET
to recover the files you were editing.

For information about the GNU Project and its goals, type C-h C-p.

GDB ~ 1980s

Copyright 2004 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are welcome to change it and/or distribute copies of it under certain conditions. Type "show copying" to see the conditions. There is absolutely no warranty for GDB. Type "show warranty" for details. This GDB was configured as "powerpc-apple-darwin"...Reading symbols for shared libraries...done

(gdb) break 2
Breakpoint 1 at 0x2ca8: file x.c, line 2.
(gdb) run
Starting program: /Users/jonathan/fse-2006-presentation/c/a.out
Reading symbols for shared libraries... done

Breakpoint 1, g (x=48) at x.c:2
 2   return x + 1;
(gdb) bt
#0  g (x=48) at x.c:2
#1  0x00002ce8 in f (y=24) at x.c:6
#2  0x00002d3c in main (argc=1, argv=0xbfff5f8 "???") at x.c:11
(gdb)
Eclipse - 2004

Visual Studio 2005
Related work

Program comprehension
  e.g., Top-down model [Soloway and Ehrlich, 1984]

Tool design based on models
  von Mayrhauser and Vans 1993
  Storey et al. 1997
Related work

How programmers do their work

Flor et al. 1991
Singer et al. 1997
DeLine at al. 2005

“Designers must first understand what it is that SEs do when they work” [Singer et al. 1997]
Research questions

What does a programmer need to know about a code base when performing a change task to a software system?

How does a programmer go about finding that information?
Research questions

**What** does a programmer need to know about a code base when performing a change task to a software system?

**How** does a programmer go about finding that information?

Contributions

1. A catalog and a categorization of 44 kinds of questions programmers ask.

2. An analysis of the process of answering questions which exposed important context for the questions.
Study approach

**Situation:** One or two programmers working on a change task

**Gather data:** Conversations between pairs of programmers and individual programmers talking aloud

**Analyze:** Identified questions asked, and worked to understand how they were answered using grounded theory
<table>
<thead>
<tr>
<th></th>
<th><strong>Study 1</strong></th>
<th><strong>Study 2</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Setting</strong></td>
<td>Laboratory</td>
<td>Industrial</td>
</tr>
<tr>
<td><strong>Participants</strong></td>
<td>9 graduate students (N1...N9)</td>
<td>16 professional programmers (E1...E16)</td>
</tr>
<tr>
<td><strong>Systems</strong></td>
<td>ArgoUML (participants were newcomers to this system)</td>
<td>Various systems that the participants were responsible for</td>
</tr>
<tr>
<td><strong>Tools</strong></td>
<td>Eclipse</td>
<td>Various...</td>
</tr>
<tr>
<td><strong>Task</strong></td>
<td>Assigned by experimenter</td>
<td>Selected by participants</td>
</tr>
</tbody>
</table>
Data collected

28 hours of audio data: 12 hours from pairs of programmers and 16 hours from individual programmers

Field notes about their use and arrangements of tools.

Example data

“Where can we get all the information” [N7]; “What can we get from here?” [N5]

“How do you get the string in here?” [N9]

“How do I get this value to here?” [E15]

“I have to somehow get the variable from the field to pass into this function” [E16]
Example data

“Where can we get all the information” [N7]; “What can we get from here?” [N5]

“How do you get the string in here?” [N9]

“How do I get this value to here?” [E15]

“I have to somehow get the variable from the field to pass into this function” [E16]

How can data be passed to (or accessed at) this point in the code? [5 sessions]
44 kinds of questions asked

Which type represents this domain concept or this UI element or action?
Which type represents this domain concept or this UI element or action?
Where in the code is the text in this error message or UI element?
Where in the code is the text in this error message or UI element?
Where is there any code involved in the implementation of this behavior?
Where is there any code involved in the implementation of this behavior?
Is there a precedent or exemplar for this?
Is there a precedent or exemplar for this?
Is there an entity named something like this in that unit (project, package or class, say)?
Is there an entity named something like this in that unit (project, package or class, say)?
What are the parts of this type?
What are the parts of this type?
Which types is this type a part of?
Which types is this type a part of?
Where does this type fit in the type hierarchy?
Where does this type fit in the type hierarchy?
Does this type have any siblings in the type hierarchy?
Does this type have any siblings in the type hierarchy?
Where is this field declared in the type hierarchy?
Where is this field declared in the type hierarchy?
Who implements this interface or these abstract methods?
Who implements this interface or these abstract methods?
Where is this method called or type referenced?
Where is this method called or type referenced?
When during the execution is this method called?
When during the execution is this method called?
Where are instances of this class created?
Where are instances of this class created?
Where is this variable or data structure being accessed?
Where is this variable or data structure being accessed?
What data can we access from this object?
What data can we access from this object?
What does the declaration or definition of this look like?
What does the declaration or definition of this look like?
What are the arguments to this function?
What are the arguments to this function?
What are the values of these arguments at runtime?
What are the values of these arguments at runtime?
What data is being modified in this code?
What data is being modified in this code?
How are instances of these types created and assembled?
How are instances of these types created and assembled?
How are these types or objects related? (whole-part)
How are these types or objects related? (whole-part)
How is this feature or concern (object ownership, UI control, etc) implemented?
How is this feature or concern (object ownership, UI control, etc) implemented?
What in this structure distinguishes these cases?
What in this structure distinguishes these cases?
What is the behavior these types provide together and how is it distributed over the types?
What is the behavior these types provide together and how is it distributed over the types?
What is the “correct” way to use or access this data structure?
What is the “correct” way to use or access this data structure?
How does this data structure look at runtime?
How does this data structure look at runtime?
How can data be passed to (or accessed at) this point in the code?
How can data be passed to (or accessed at) this point in the code?
How is control getting (from here to) here?
How is control getting (from here to) here?
Why isn’t control reaching this point in the code?
Why isn’t control reaching this point in the code?
Which execution path is being taken in this case?
Which execution path is being taken in this case?
Under what execution circumstances is this method called or exception thrown?
Under what execution circumstances is this method called or exception thrown?
What parts of this data structure are accessed in this code?
What parts of this data structure are accessed in this code?
How does the system behavior vary over these types or cases?
How does the system behavior vary over these types or cases?
What are the differences between these files or types?
What are the differences between these files or types?
What is the difference between these similar parts of the code (e.g., between sets of methods)?
What is the difference between these similar parts of the code (e.g., between sets of methods)?
What is the mapping between these UI types and these model types?
What is the mapping between these UI types and these model types?
Where should this branch be inserted or how should this case be handled?
Where should this branch be inserted or how should this case be handled?
Where in the UI should this functionality be added?
Where in the UI should this functionality be added?
To move this feature into this code what else needs to be moved?
To move this feature into this code what else needs to be moved?
How can we know this object has been created and initialized correctly?
How can we know this object has been created and initialized correctly?
What will be (or has been) the direct impact of this change?
What will be (or has been) the direct impact of this change?
What will be the total impact of this change?
What will be the total impact of this change?
Will this completely solve the problem or provide the enhancement?
Will this completely solve the problem or provide the enhancement?
e.g., a method

e.g., a calls relationship
1. Finding initial focus points
   5 questions

2. Building on focus points
   15 questions

3. Understanding a subgraph
   13 questions

4. Questions over groups of subgraphs
   11 questions
1. Finding initial focus points
   e.g., Which class represents this domain concept or this UI element? [7 sessions]

2. Building on focus points

3. Understanding a subgraph

4. Questions over groups of subgraphs
1. Finding initial focus points

2. Building on focus points
e.g., Where is this variable or data structure being accessed? [8 sessions]

3. Understanding a subgraph

4. Questions over groups of subgraphs
1. Finding initial focus points

2. Building on focus points

3. Understanding a subgraph
e.g., What in this structure distinguishes these cases? [3 sessions]

4. Questions over groups of subgraphs
1. Finding initial focus points

2. Building on focus points

3. Understanding a subgraph

4. Questions over groups of subgraphs
   e.g., What is the difference between these similar parts of the code? [8 sessions]
Answering questions

Question ➔ Supporting question ➔ Tool question ➔ Result set
Relationships between questions

For example, questions and sub-questions.

“Trying to take my questions and filter those down to something meaningful where I could take a next step.” [N4].
Questions and tools

Mismatch between tools and questions and non-optimal choice of tools.

“Which classes have MEvents as fields?” [N3]
From results to answers

Multiple result sets, often noisy relative to the intended question.

Often these results are often presented separately (can’t be seen together).
From results to answers

Multiple result sets, often noisy relative to the intended question. Often these results are often presented separately (can’t be seen together).

“I was starting to forget who was calling what, especially because there is only one search panel at a time that I can see” [N6]

“You go down a path to try to find out some information and it leads to a dead end and you got to start all over again.” [E16]
Limitations and open questions

Generalizable?

• Types of software change tasks
• Programming tools our participants used

Open questions

• Do questions change with experience?
• Which questions are most important for tools to support?
• Which strategies for answering questions are most effective in the long run?
Research questions

What does a programmer need to know about a code base when performing a change task to a software system?

How does a programmer go about finding that information?

Contributions

1. A catalog and a categorization of 44 kinds of questions programmers ask.

2. An analysis of the process of answering questions which exposed important context for the questions.