Usage of Program Analysis Tools

Dear Participant:

My name is Emerson Murphy-Hill, and I am a student at Portland State University. I am beginning a study on program analysis tools, and would like to invite you to participate. You are being asked to take part because of your programming experience.

As part of the study, I am interested in how you use programming tools, and hope that the information I collect will help us to better understand these tools. If you decide to participate, you will be asked to use tools that you may or may not have used in the past. I will observe how you use the tools, and may ask you questions as we go along so that I understand what you are doing and why. You may not receive any direct benefit from taking part in this study, but the study may help to increase knowledge that may help others in the future.

Any information that is obtained in connection with this study and that can be linked to you or identify you will be kept confidential. Subject identities will be kept confidential by not recording any personal information.

Participation is entirely voluntary. Your decision to participate or not will not affect your relationship with the researcher or with Portland State University in any way. If you decide to take part in the study, you may choose to withdraw at any time without penalty. Please keep a copy of this letter for your records.

If you have concerns or problems about your participation in this study or your rights as a research subject, please contact the Human Subjects Research Review Committee, Office of Research and Sponsored Projects, 111 Cramer Hall, Portland State University, (503) 725-4288. If you have questions about the study itself, contact Emerson Murphy-Hill at Department of Computer Science, P.O. Box 751, Portland State University, Portland, Oregon 97207-0751, (503) 725-4036.

Sincerely,
Emerson Murphy-Hill
Graduate Student, Portland State University
Usage of Program Analysis Tools

Test Setup

Date: _____ Participant Number: _____

Material Checklist

- ☐ Letter
- ☐ This Packet (6 pages)
- ☐ 2 Questionnaires
- ☐ Lined notebook
- ☐ Smell cards in envelope, ordered, front/back
- ☐ Clipboard
- ☐ 2 Pens
- ☐ Mouse
- ☐ Laptop
- ☐ AC Power Supply
- ☐ Watch
- ☐ Snack
- ☐ IEEE Software
- ☐ ~~~

Set up

- ☐ Mark the tool order AB BA
- ☐ Reshuffle packet, staple
- ☐ Mark the code order 1 2
- ☐ Make sure all Bookmarks are visible, available as fast view
- ☐ Open ToolDemo, Full Screen at 1280x800
- ☐ Show line numbers
- ☐ "Show Single Element Only" is off
- ☐ Highlights are off
- ☐ Sitting to the right of subjective
- ☐ Hand participant letter

Tear Down

- ☐ Take participant's survey, loose interview
- ☐ Thank participant, release
- ☐ Make sure you've got a time and a date
- ☐ Close all editors, except Demo
- ☐ Jot down all mentally queued observations
- ☐ Transcribe notes

Pre-Experiment Questionnaire

The following questionnaire is intended for us to get an idea of what your programming background is. Your answers in no way affect the rest of the experiment, it simply gives us context for interpreting the result.

Feel free to write in the margins to explain your answers, if necessary.

Job title: _____

How many years have you been programming? _____

Over the last year, about how many hours per week would you say you spend programming, on average? _____

How proficient, on a scale from 0 to 4, where 0 means "not at all" and 4 means "expert"?

Java	*0*	*1*	*2*	*3*	*4*
C++	*0*	*1*	*2*	*3*	*4*

When programming, do you typically use an Integrated Development Environment? *Y / N*
If so, which one(s) and for what % time? _____
What non-IDE editors do you use for programming? _____

On a scale form 0 to 4, how familiar are you with the practice of refactoring?

(0 = not at all, to 4 = very familiar)	*0*	*1*	*2*	*3*	*4*

Do you use any refactoring tools? *Y / N* If so, which ones?

_____

On a scale form 0 to 4, how familiar are you with code smells?

(0 = not at all, to 4 = very familiar)	*0*	*1*	*2*	*3*	*4*

Please Hand This Back to Experimenter

Experimental Procedure

Introduction

What we're going to do in this experiment is investigate code smells, which were originally proposed in Martin Fowler's book on refactoring. The idea is that smells help you identify candidates for refactoring; for instance, the "Long Method" smell suggests that you should perhaps perform the Extract Method refactoring. You needn't be too familiar with the concept; we'll do some review as we go along and you are free to ask questions.

This experiment will have four parts:

[AB] In the first part, I'll ask you about smells in code. In the second part, I'll give you a tool to help find smells.
In the third part, I'll ask you some details about smells, and in the fourth part, I'll ask you about the details with the assistance of the tool.

[BA] In the first part, I'll ask give you a tool to help find smells. In the second part, I'll ask you about smells without the help of a tool.
In the third part, I'll ask you some details about smells with the help of the tool, and in the fourth part, I'll ask you about the details without the tool.

In a moment, I'll give you eight 3 by 5 cards. Each card will have the name of a code smell and its definition, with an example on the back. I'll give you a few minutes to read them, then you'll give them back to me when you're ready, and then we'll begin looking at some code. Questions?

Give the participant the card stack, and await their completion. If it takes more than a few minutes, ask them if they are finished. If they are not satisfied within 10 minutes, tell them that we'll move on regardless, and that you'll make a note that they were not finished.

Ok, so now we're going to look at some code. As you work, please don't modify the code, or navigate outside of the editor. As a rule of thumb, please try to spend no more than 3-5 minutes per file.

[A] Manual Finding

Take cards back. Now I'll ask you to look at a Java file and try to spot some of the code smells that you saw on the cards. You'll scroll through one Java file, while skimming the code top to bottom. If you see an interesting smell, just say so out loud.

Open up ToolDemo. So for instance, you would scroll through this file from top to bottom, noting any smells you notice.

Questions?

[1] Open scroll1.		**[2]** Open scroll3.
Data Clumps Feature Envy Message Chains Switch Statements Typecast Instanceof Large Method Large Class		
[1] Open scroll8.		**[2]** Open scroll7.
Data Clumps Feature Envy Message Chains Switch Statements Typecast Instanceof Large Method Large Class		

[B] Tool Finding

Take cards back. I'll ask you to spot some of the code smells that you saw on the cards. You'll scroll through one Java file, while skimming the code top to bottom, with the help of a smell detection tool.

(Open up ToolDemo, activate tool) The tool is represented by a visualization behind your java code. *(Scroll).* It looks a bit like a bunch of petals on a flower. Each petal represents a smell, and we can hover over to see the name of the smell *(demo)*. The size of the petal represents how bad that smell is in the code that you are looking at. As this tripwire passes over methods *(demo)*, or when the cursor is in a method, the smells for that method are visualized.

This part of the tool is intended to give you an idea of which smells are present. There's more detail to the tool, but we'll get to that later.

So, the task is, if the tool helps you see an interesting smell, just say so out loud. Ready?

[1] Open scroll3.		**[2]** Open scroll1.
Data Clumps Feature Envy Message Chains Switch Statements Typecast Instanceof Large Method Large Class		
[1] Open scroll7.		**[2]** Open scroll8.
Data Clumps Feature Envy Message Chains Switch Statements Typecast Instanceof Large Method Large Class		

[A] Manual Finding

(Switch to full screen editor)
Now what we'll do is look at one code smell in depth; Feature Envy. *(Open up ToolDemo)* Suppose I analyze this for Feature Envy by inspection.

Looking at this detail, I might conclude that the method, or some parts of it, should be moved to DHTTransportFullStats.

So the task that I want you to do is to make some judgments about the code; how widespread the Feature Envy is, how likely you are to remove it, and how you might do it. I'll ask these questions as you work, and if you have any questions for me, feel free to ask. When you're finished, let me know.

Any questions? *(Pause)* Ok, give it a try on this method.

[1] Open envy8.	**[2]** Open envy3.
How widespread is the smell?	
How likely are you to remove it?	
How might you remove it?	

[1] Open envy1.	**[2]** Open envy6.
How widespread is the smell?	
How likely are you to remove it?	
How might you remove it?	

[B] Tool Inspection

(Switch to full screen editor)
Now what we'll do is look at one code smell in depth; Feature Envy. *(Open up ToolDemo)* Suppose that I glance at the smell indicator and see that Feature Envy is high. I can then click on its label *(do it)*, and get a detailed view of what's going on.

The movable sheet shows me which classes members are referenced, and assigns each class a color. So for instance *(point)*, I can see that many members of DHTTransportFullStats are referenced, but only one member in this class is referenced. The associated members are highlighted in source code, and I can mouse-over the classes and members to emphasize their occurrences in code.

Looking at this detail, I might conclude that the method, or some parts of it, should be moved to DHTTransportFullStats.

So the task that I want you to do is to use the tool to help you make some judgments about the code; how widespread the Feature Envy is, how likely you are to remove it, and how you might do it. I'll ask these questions as you work, and if you have any questions for me, feel free to ask. When you're finished, let me know.

Any questions? *(Pause)* Ok, give it a try on this method.

[1] Open envy3.	**[2]** Open envy8.
How widespread is the smell? How likely are you to remove it? How might you remove it?	
[1] Open envy6.	**[2]** Open envy1.
How widespread is the smell? How likely are you to remove it? How might you remove it?	

Post-Experiment Questionnaire

Please answer a few questions about your experience during this experiment. Feel free to write comments in the margins. We just saw a tool to assist in the detection and understanding of code smells. Below are two questions about several characteristics that such a tool might have.

Characteristic	How important is the characteristic to *any smell detection tool*?						Do you agree that the characteristic *applies to the tool you just used*?				
	Not Important	Somewhat Important	Important	Very Important	Essential		Strongly Disagree	Somewhat Disagree	Neutral	Somewhat Agree	Strongly Agree
Example: The tool should help me cook dinner.	☐	☐	☐	☒	☐		☒	☐	☐	☐	☐
Example: The tool should use pretty colors.	☒	☐	☐	☐	☐		☐	☐	☐	☒	☐
The tool should not distract me.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐
The tool should have a user interface consistent with the rest of the environment.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐
The tool should make smell information available to me at all times.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐
The tool should tell me first and foremost about smells related to the code I'm working on.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐
The tool should not block me from my other work while it analyzes or finds smells.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐
The tool should emphasize smells that are difficult to see with the naked eye.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐
The tool should not overwhelm me with the smells that it detects.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐
In addition to finding smells for me, the tool should tell me *why* smells exist.	☐	☐	☐	☐	☐		☐	☐	☐	☐	☐

When showing me details about code smells, the tool should show me the relationships between effected program elements.	☐ ☐ ☐ ☐ ☐		☐ ☐ ☐ ☐ ☐
The tool should help me estimate the extent of a smell in the code.	☐ ☐ ☐ ☐ ☐		☐ ☐ ☐ ☐ ☐
The tool should help me decide whether to remove a smell from the code.	☐ ☐ ☐ ☐ ☐		☐ ☐ ☐ ☐ ☐

Please state whether you agree with the following statements:

	Strongly Disagree	Somewhat Disagree	Neutral	Somewhat Agree	Strongly Agree
The smell detector that I used in this experiment was *useful for the given tasks*	☐	☐	☐	☐	☐
The detector found information that I would not have found *as quickly* without it.	☐	☐	☐	☐	☐
The detector found information that I would not have found *at all* without it.	☐	☐	☐	☐	☐
Without the tool, it was difficult to look for all 8 smells at the *same time*.	☐	☐	☐	☐	☐
If a detector like the one in this experiment were available, *I would use it* when I code.	☐	☐	☐	☐	☐

Please Hand This Back to Experimenter

Loose Interview

If you were using this tool while coding, do you think that it would get your attention at the right times?

Would it be too distracting?

Did the tool make you more confident about your refactoring judgements, with respect to feature envy?

Do you think it helped you make more informed judgments?

If you could *change something* about the smell detector, what would it be?