How you should (not) do a user study & Development of research "prototypes"

http://d3s.mff.cuni.cz

Lukáš Marek
lukas.marek@d3s.mff.cuni.cz
Motivation of this talk

This talk is not about “golden rules”

We will let you to learn from our mistakes and hopefully give you some useful tips
DiSL usability study – with test subjects

work of Aibek Sarimbekov
We wanted to show, that development of instrumentation with DiSL is easy – easier than with other tools

(Sadly, some people out there still don't know this...)}
User study

- Get the subjects
- Prepare case study tasks
- Brief subjects
- Separate them into groups
  - Maintain validity of the study
- Do the user study
- Evaluate the results
Our 1\textsuperscript{st} attempt

We did the evaluation and results were “exactly” opposite as we wanted them.
What went wrong?

- We were too “objective” and compared not only the strong sides of our tool
- We were too confident and were not thinking about the results
What now?

Find other subjects and do the experiment again

(Swap the groups)
Our 2\textsuperscript{nd} attempt – details

- 16 BSc., MSc., and PhD students
- Six instrumentation tasks for DiSL and ASM
- Tutorial on JVM internals, ASM and DiSL
- Separation according to the results from questionnaire
  - Expertise in OOP, Java, Eclipse, ASM, AOP, ...
- Initial training example
- The user study evaluated
  - Time to complete each task
  - Correctness of each task (0-4 points)
User study results – intro

- Each task implemented in
  - DiSL – instrumentation framework
  - ASM – bytecode manipulation library

- Evaluated characteristics (aggregated as average)
  - Time to complete the task
  - Correctness of the task
Task 1

- On method entry, print the number of method arguments

<table>
<thead>
<tr>
<th>Group</th>
<th>Time (min)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>26</td>
<td>2.0</td>
</tr>
<tr>
<td>DiSL</td>
<td>7</td>
<td>3.3</td>
</tr>
</tbody>
</table>
Task 2

- Before array allocation, invoke a given method that receives the array length as its argument

<table>
<thead>
<tr>
<th>Group</th>
<th>Time (min)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>13</td>
<td>2.4</td>
</tr>
<tr>
<td>DiSL</td>
<td>9</td>
<td>2.6</td>
</tr>
</tbody>
</table>
Task 3

Upon method completion, invoke a given method that receives the dynamic execution count of a particular bytecode instruction as its argument.

<table>
<thead>
<tr>
<th>Group</th>
<th>Time (min)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>19</td>
<td>2.4</td>
</tr>
<tr>
<td>DiSL</td>
<td>12</td>
<td>3.0</td>
</tr>
</tbody>
</table>
Task 4

Before each AASTORE bytecode, invoke a given method that receives the object to be stored in the array together with the corresponding array index as its arguments.

<table>
<thead>
<tr>
<th>Group</th>
<th>Time (min)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>29</td>
<td>0.4</td>
</tr>
<tr>
<td>DiSL</td>
<td>8</td>
<td>3.0</td>
</tr>
</tbody>
</table>
Task 5

- On each INVOKEVIRTUAL bytecode, invoke a given method that takes only the receiver of the invoke bytecode as its argument

<table>
<thead>
<tr>
<th>Group</th>
<th>Time (min)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>30</td>
<td>0.0</td>
</tr>
<tr>
<td>DiSL</td>
<td>10</td>
<td>3.0</td>
</tr>
</tbody>
</table>
On each non-static field write access, invoke a given method that receives the object whose field is written to, and the value of the field as its arguments. Invocation shall be made only when writing non-null reference values.

<table>
<thead>
<tr>
<th>Group</th>
<th>Time (min)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASM</td>
<td>30</td>
<td>0.6</td>
</tr>
<tr>
<td>DiSL</td>
<td>8</td>
<td>3.8</td>
</tr>
</tbody>
</table>
User study – summary

DiSL proved to be more effective tool for instrumentation development than ASM
Development of research prototypes
How far should we push it?

Do we want users for our software?
YES !!!
Really ?
Why not

- Initial setup of your project should be shorter than the usual 30 hours
- Code is not self-documenting
- There is always something you can improve
- There is always something you should fix
- Devil is in the detail

- The users start to want something...
- ...but nobody will help you (at least initially)
- Development will not give you PhD
- You can (will) get frustrated
DiSL as a software prototype

- In development for 2 years now
- A few thousands lines of code
- Relatively tested on several case studies
- A few users
- Some parts are even documented
Why we manage it?

- Initially two developers
  - Keeps you motivated
- Other guys were doing the testing
  - Protects from frustration
  - Gives you feedback
  - Bug fixing is still on you
- Bigger testing examples can be used for an evaluation part of a paper
  - You need to know the evaluation examples sooner than one week before the paper deadline
- Almost all the documentation is (will be) published
  - Tutorial papers (even journals)
Why should I bother then?
I'm not saying anything new here. This is a long standing problem. Just to discuss it a bit.
Personal experience

- We needed evaluation for ShadowVM
- We found six publications that included description of an existing analysis that would be a good candidate for re-implementation
  - Some published in
    - IEEE Transactions on Software Engineering
    - OOPSLA
    - ICSE
  - Some published by
    - University of California, Berkeley
    - University of California, Irvine
    - IBM T. J. Watson Research Center
- Only one publication contained (non-working) link to additional materials
- We asked three of them and received one positive reply
Why not providing a prototype?

- Prototype is in a state, where it is not usable by anyone else
- Prototype (software) is confidential
- It can introduce additional work
By providing a prototype

- Reviewers can see, that you are not making things up
- Others can easily build on your work
- Development and evaluation gives you bunch of new problems you could not see from a drawing board
Why not provide anything you already have?