Introduction

I’m here to understand better how we can integrate multiple static analysis tools in a single interface, particularly about showing bugs found for a same codebase by different analysis tools, feedback to know bug fix is ongoing and traceability of bug fixing. This interview will take about 60 minutes, during which time we’ll go through some questions. Throughout, I’d like you to treat me as if you’re describing the situation to someone who is developing the tool for you. I’m here to learn from you.

A couple of things before we start. To the extent possible, I will take your comments to be confidential. I will aggregate all the comments from several interviews I’m conducting so that your comments are not easily traced to you. If I quote you in my final report, I will do so without identifying your name or specific role. If there’s anything you really don’t want on the record, even if it’s anonymized, please let me know that, too. Also, this interview is entirely voluntary on your part – if for any reason you want to stop, please let me know. We can end the interview at that point with no repercussions for you of any kind. I can also throw out anything you’ve told me until that point.

Do you have any questions for me? All right, then, let’s proceed.

I shall start recording now as mentioned in consent form. This is just so that I don’t miss anything – no one other than me and if necessary, my thesis advisors will have access to the recording. Thanks.

Pre-test Questionnaire:

Q. How often do you do software development (i.e., coding)?

- Is it daily, weekly etc?

Q. Have you used static analysis tools?

- tool to find bugs in your code!

Q. What tools have you used?

- Is it IDE integrated tool or any other dedicated tools like FindBugs, PMD etc.

- Do you ever try to use different tool based? If no, why?

If yes, why? And what challenges did they face while using tools separately?

[Crosscheck whether our UI RQs address those challenges and if not, open them for improvisation]

Q. Which is your favourite one?

Q. Why it is your favourite?

- Any correlation to its better user interface feature?

Assume you are working on a project and want to find bugs in your code. There are ten tools linked to your codebase to have better coverage of vulnerabilities. Now let us walkthrough 3 main questions with respect to its user interface.

[Research Question 1]

The first one;

Q. How to display the results of the same codebase from different analysis tools?

(present the prototypes one after other in random order, do cognitive walkthrough – think aloud)

Scenario: Assume you as a Software Developer working on a project called "Alpha". On next working day, you are about to see analysis results from multiple tools and your primary intention is to make your code base bug free.

Task: What are the bugs reported for file XSSFilter.java?

Solution Ideas - Prototype [list, next]:

Task: What are the bugs reported for file XSSFilter.java?

> **Success Criteria**:
> User reports the names of bugs i.e., XSS_Config, JSP reflected vulnerability and HttpSession.

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with for given task?

- easy to use?

Q. Why?

Q. Rate: perceived usability [O be low, 10 be high] – both designs in comparison

Solution Ideas - Prototypes [next, horizontal]:

Success Criteria:
User reports the names of bugs i.e., XSS_Config, JSP reflected vulnerability and HttpSession.

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with for given task?

- easy to use?

Q. Why?

Q. Rate: perceived usability [O be low, 10 be high] – both designs in comparison

Solution Ideas - Prototype [horizontal, vertical]:

Task: What are the bugs reported for file XSSFilter.java?

Success Criteria:
User reports the names of bugs i.e., XSS_Config, JSP reflected vulnerability and HttpSession.

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with for given task?

- easy to use?

Q. Why?

Q. Rate: [O be low, 10 be high] – both designs in comparison

Q. Do you prefer to know the tool names at bug reports in context of code view?

- Why?

Solution Ideas - Prototype [horizontal, table]:

Task: What are the bugs reported for file XSSFilter.java?

Success Criteria:
User reports the names of bugs i.e., XSS_Config, JSP reflected vulnerability and HttpSession.

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with for given task?

- easy to use?

Q. Why?

Q. Rate: [O be low, 10 be high] – all designs in comparison

Scenario: In same scenario as so far, while looking at bug info at a certain code line, now you would like to see similar bugs in the code file so as to overcome issues with tools reporting wrong line numbers which reporting same bug by other tool or just for information.

Solution Ideas - Prototype [similar boxes, similar list]:

Task: What are the similar bugs reported in file XSSFilter.java?

Success Criteria:
User reports the names of bugs i.e., XSS_Config, JSP reflected vulnerability.

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with for given task?

- easy to use?

Q. Why?

Q. Rate: [O be low, 10 be high] – all designs in comparison

[Research Question 2]

Let's switch to second main question;

Q. What feedback works to know that the bug fixing is on-going?

Scenario: Assume you as a Software Developer working on a project. The project is integrated with multiple tools independently with native user interfaces i.e., CLI, IDE and WEB based. In contrast you have the project with unified interface for them.

(present the project integrated with WEB, IDE and CLI based are SonarCloud, IntelliJ + SonarLint and ESLint tools respectively…

Next present the merged prototype with 5 feedbacks which investigated earlier are animated icon. progress bar, status pending popup, status spinner, alert box.,

then do cognitive walkthrough)

Task: Identify the bugs in test.js file using CLI, IDE and WEB based analysis tools?

Success Criteria:
User reports the names of bugs shown from each interface.

Task: Try to fix the bug of any unused-variable in index.js and see if bug is fixed or not?

Success Criteria:
User reports whether it got fixed or not.

Follow up:

Feedback features while working on a project integrated with multiple tools in context of unified interface against native user interfaces.

Rating: [0 (no such feature, worst) to 10 (best)]

- User perceived usability rating for MSAT-UI in comparison to having independent tools and vice-versa.

Q. How usable is animated icon feature.

Q. How usable is progress bar feature.

Q. How usable is status pending popup feature.

Q. How usable is status spinner feature.

Q. How usable is alert box feature.

Next,

Q. Do you prefer alert box feedback when bug fix is failed which is absent in existing tools?

- Why?

Q. Do these 5 feedback features with unified user interface help in fixing more bugs (perceived usability)?

- Why?

Q. Do these 5 feedback features with unified user interface help in fixing bugs faster (perceived usability)?

- Why?

[Research Question 3]

Let's switch to final main question;

Q. How to carry traceability of bug fixing?

(Explain what traceability in this scenario is.)

(Here in this scenario traceability mean to see how each attempt / commit to fix a bug effects the metrics of the analysis tools. This ensures some safety to prevent new bugs.)

Scenario: In same scenario as so far, user notices there are some bugs fixed earlier in the same method. He/she would like to know what they are and so this might help in maintaining the method to be bug free by not re-introducing old bugs again.

Solution Ideas - Prototype [multiple, table]:

Task: Identify how many bugs are fixed earlier in **stripXSS** method in file **XSSRequestWrapper.java**?

Success Criteria:
User reports the number of bugs i.e., 2.

Task: What is the bug line reported as **SA** Bug type earlier in **stripXSS** method in file **XSSRequestWrapper.java**?

> **Success Criteria**:
> User reports the line as “L44: value = value.replaceBy("";"");”.

Follow up:

Q. Among the 2 solution ideas presented, which one do you feel convenient with for given tasks?

- easy to use?

Q. Why? [Do you prefer having multiple windows to single window?]

Q. Rate: [0 be low, 10 be high] – both designs in comparison

Q. Which UI seems more scalable in this context of traceability helping you in state of workflow?

Q. Why?

Conclusion

Thank you – those are all the questions I have for you. If anything, else occurs to you later, please don’t hesitate to let me know by email. Do you have any questions? Thanks again! for your participation and helping me understand.

This protocol answers the selected research questions. It includes questionnaire that provides data for qualitative and quantitative analysis. In terms of quantitative, the metrics considered are the perceived usability in comparison to other solution idea, the time and steps needed to do the task and whether the user succeeds in doing the task or not. And for qualitative analysis, users reasoning is considered as formative feedback.