

Integration of Multiple Static Analysis Tools in a Single Interface

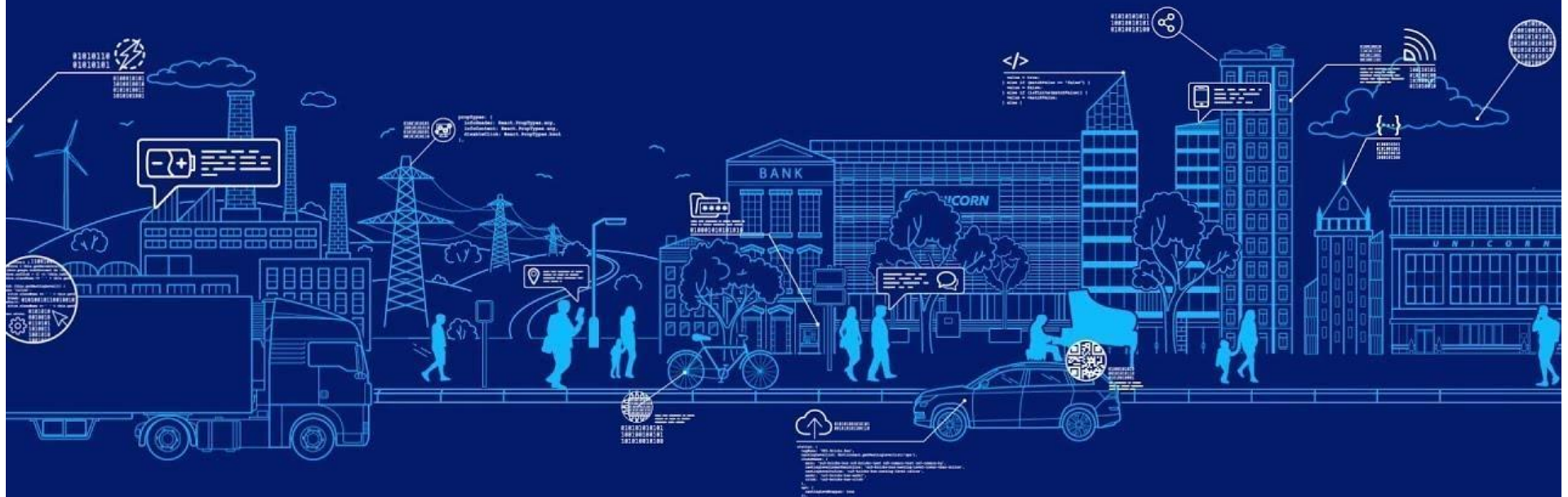
- Author

- G. S. Varma

- Supervisors:

- Prof. Dr. Eric Bodden
- Dr.-Ing. Ben Hermann

Software Everywhere



- “ **\$1.1 Trillion** in Assets Affected by Software Bugs in **2016** “

- Software Fail Watch Annual Report,

[Tricentis](#)



Static Code Analysis

- It helps in prevention of bugs.
- It examines code without execution.
- Detects Vulnerabilities :
 - Injections
 - Cross Site Scripting (XSS)
 - Buffer Overflow, and Dead Code etc



Static Code Analysis

■ Tools :

- IDE Notifications,
- IDE tools,
- Dedicated tools,
- Linters
- CLI tools.

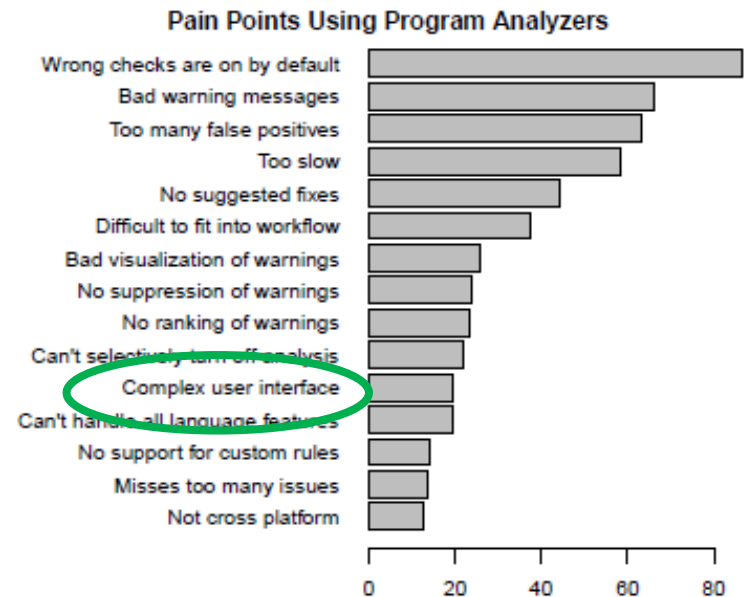


■ Research Papers:

- Christakis, Maria; Bird, Christian (2016): What developers want and need from program analysis: an empirical study.
- Johnson, Brittany; Song, Yoonki; Murphy-Hill, Emerson; Bowdidge, Robert (2013): Why don't software developers use static analysis tools to find bugs?

■ Found: developers facing issues in using tools

■ Most importantly, **USABILITY** issue.



SARIF



- Static Analysis Results Interchange Format (SARIF)
- Standard representation of bug warnings in a JSON format

Multiple Tools

- Developers seem to use multiple static analysis tools each having own coverage.
- Research trends:
 - Using multiple static analysis tools in order to prioritise the bug warning alerts
 - Using results of three different static analysis tools for a programming language, Java and merges them together in order to show warnings to the developer

But **USABILITY** is not addressed...

Multiple Tools

- **SARIF** scope - different analysis tools results can be integrated
- **Need** for addressing **Usability** issue

Thesis Work Plan

- Problem Statement
- Research Questions
- What Current Tools do?
- Our Approaches
- Evaluation
- Time Plan

Problem Statement

- How to integrate the results of multiple static analysis tools

in a unified user interface?

Research Questions

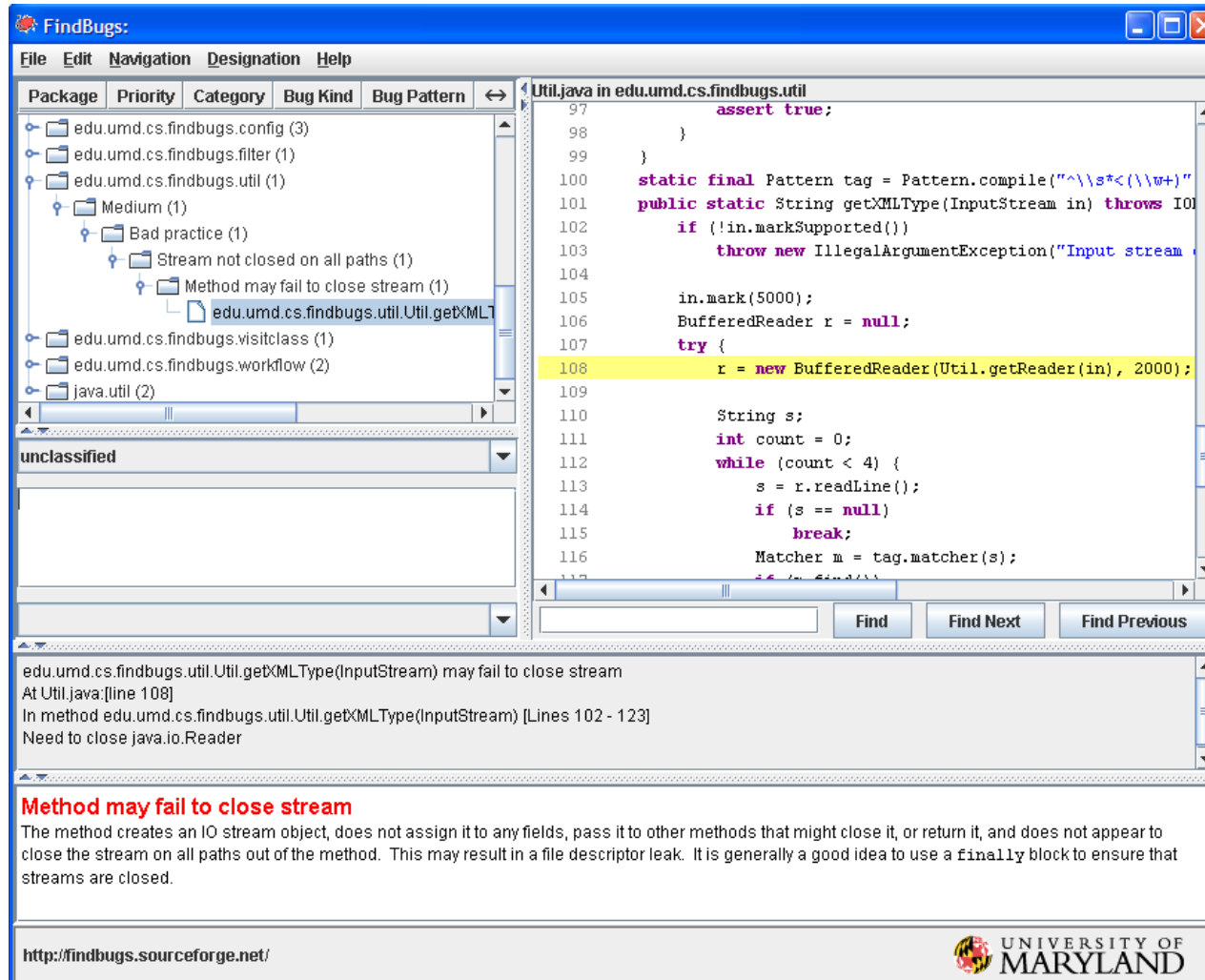
1. How to display results of the same codebase from different analysis tools?
2. What feedback works to know that the bug fixing is on-going?
3. How to carry traceability of bug fixing?

Thesis Work Plan

- Problem Statement
- Research Questions
- What Current Tools do?
- Our Approaches
- Evaluation
- Time Plan

What Current Tools do? - RQ 1

FindBugs



What Current Tools do? - RQ 2

- FindBugs

New Project

Project name (i.e., description)

Class archives and directories to analyze

Add

Remove

Auxiliary class locations

Add

Remove

Source directories

Add

Remove

Finish Cancel

What Current Tools do? - RQ 3

■ TeamScale



Added db2 database mapping after reading forum post

by [Daniel Lewis](#) in revision [91687a1146419dd23ceaed299185512696643dc1](#) (git)

Files: 11 changed

Findings: 0 4 12 1

Jul 17 2014 10:53



Add getDelegationState() in DelegateTask.

by [Anya Hill](#) in revision [812b1e277d844fa48307bcd7c692a6f395c85fbb](#) (git)

Files: 14 changed

Findings: 0 3 12 5

Jul 17 2014 10:30



TASK_TIMEOUT

by [Jacob Nelson](#) in revision [997da57af6f2c08d504473d3e9837788b7592dcb](#) (git)

Files: 14 changed

Findings: 0 5 12 3

Jul 17 2014 08:46

Thesis Work Plan

- Problem Statement
- Research Questions
- What Current Tools do?
- Our Approaches
- Evaluation
- Time Plan

Our Approaches



Our Approaches

- Research Methodology – UX Design Cycle
- Software Engineering disciplines:
 - Complex datasets
 - Compiler reporting
 - Continuous integration
 - Refactoring tools
 - Issue tracker
 - Stack Overflow
 - Gamification
 - Usability Engineering

Our Approaches

- Complex datasets:

- Dix et. al. - more complex grouping and linking of datasets in the context of a user interface of Spreadsheets application.

Design lesson : extensibility of columns



- Gaur et. al.

- linear search problem in indexing as it takes more time for large volumes of data.
So, different parameters are introduced to decrease computation time.

Example: Searching for toy



Our Approaches

- Compiler reporting

Horning et. al

- error logging with statistics
- stating what kind of bugs are not found along with bugs found



Our Approaches

■ Refactoring tools

Dustinca

- barrier of discoverability
- introduced a smart tag for code can be refactored.
- 'on-board' phase _ **Gamification**



Hayashi et. al. - task level commits in order to maintain edit history



Our Approaches

- Issue tracker

Baysal et. al. :

- Information overload
- Expressiveness

Ideal to describe the priority as per team decision instead of personal choice.



Our Approaches

■ Stack Overflow

- Wang et. al. : 10934198 questions on a 'User Interface' topic
- Treude et. al. : 72.30 % questions have between 2 and 4 tags



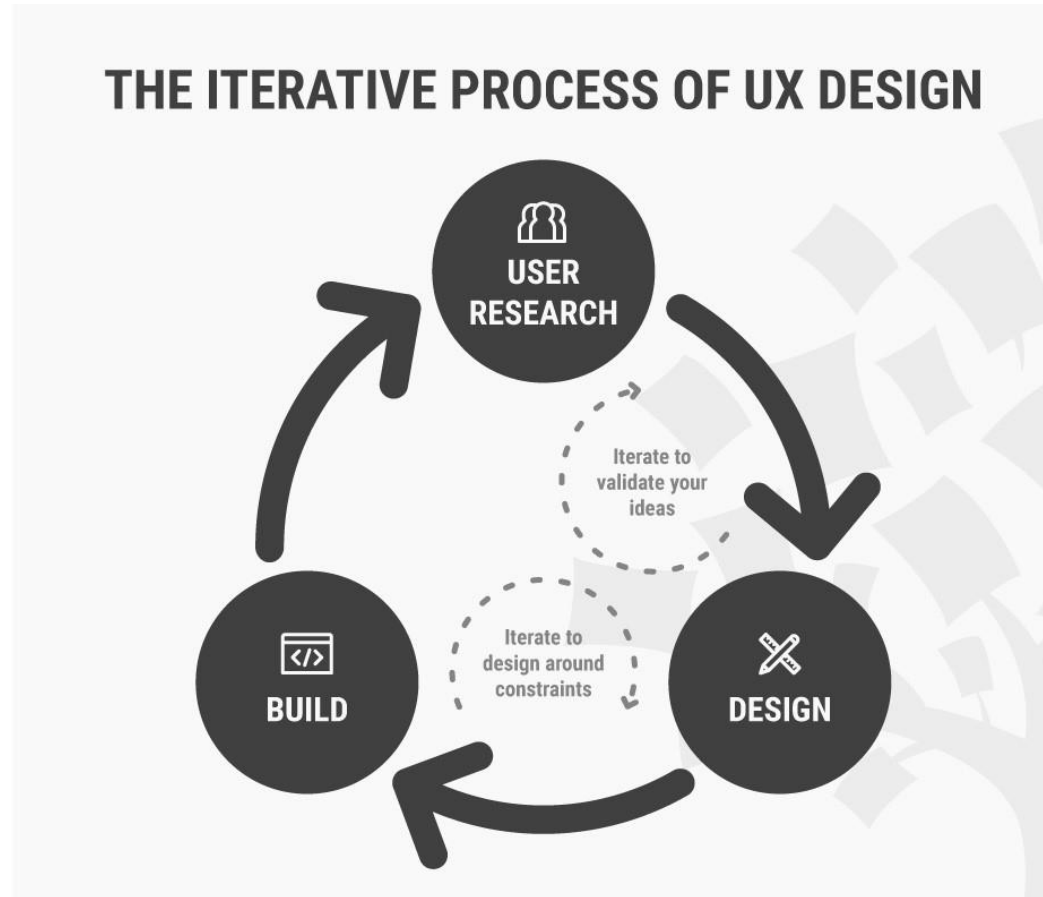
The screenshot displays the Stack Overflow website interface. At the top, there is a search bar and navigation links for 'Log In' and 'Sign Up'. The left sidebar contains a 'Home' section with links to 'Stack Overflow', 'Tags', 'Users', and 'Jobs', as well as a 'Teams' section with a 'Learn More' button. The main content area is titled 'Top Questions' and features a list of questions with their respective statistics (votes, answers, views) and tags. The questions listed are:

- Text-To-Speech in Android Studio That Works Offline** (0 votes, 1 answer, 9 views) with tags: java, android, android-studio, text-to-speech. Answered 37 secs ago by Benjamin Hue 28.
- How to set adaptive constraints the right way** (0 votes, 0 answers, 2 views) with tags: ios, swift, constraints. Asked 41 secs ago by berrys 8.
- Push to array in nested immutable object** (0 votes, 0 answers, 2 views) with tag: immutable.js. Asked 50 secs ago by Nuru Salihu 2,263.
- How to get a text direction given a LatLng?** (0 votes, 0 answers, 2 views) with tags: android, google-maps, android-mapview, android-maps-v2, android-maps. Asked 52 secs ago by NullPointerException 13.5k.

On the right side, there is a 'Hot Network Questions' section with a list of trending questions, including:

- Why Did Howard Stark Use All The Vibranium They Had On A Prototype Shield?
- How to deal with fear of taking dependencies
- Patience, young "Padovan"
- Information to fellow intern about hiring?
- What is the steepest angle that a canal can be traversable without locks?
- What does "rabbited" mean/imply in this sentence?
- Does light intensity oscillate really fast since it is a wave?
- Is it legal to have the "© (c) 2019 John Smith" header in all files when there are hundreds of contributors?
- Landlord wants to switch my lease to a "Land contract" to "get back at the city"
- I see my dog run
- description of papers that have not been submitted to a venue?
- The difference between dialogue marks

UX Design Cycle



Example: RQ 1

■ Prototype 1

The screenshot shows the 'D SAT Interface' window. At the top, it says 'Project: Alpha'. Below this is a table with the following columns: Name (Bug title), Tool, Type, Fix Location, and Assignee. The table contains five rows of bug data. The first row, 'FI_EMPTY', is highlighted in blue. To the right of the table is a 'Filters' sidebar with sections for 'Tool' (checkboxes for 'toolLong' and 'toolShort'), 'Bugs' (checkboxes for 'My Bugs' and 'All Bugs'), and 'Vulnerability Type' (checkboxes for 'SQL Injection' and 'XSS'). Below the table is a 'Bug Description' section for the selected 'FI_EMPTY' bug, containing text about deleting empty finalizers and two buttons: 'Fix Now' and 'Know More'.

Name (Bug title)	Tool	Type	Fix Location	Assignee
FI_EMPTY		FI	12.4 EditList.java	Varma
EQ_CHECK		EQ	6.3 LoopHelper.java	Max
CO_SELF		CO	11.2 StringComparer.java	Un-assigned
XSS_REQUEST		XSS	5.4 HttpSender.java	John
DMI_EMPTY		DM	3.3 DatabaseHelper.java	Elina

Bug Description

FI: Empty finalizer should be deleted (FI_EMPTY)

Empty **finalize()** methods are useless, so they should be deleted.

[Fix Now](#) [Know More](#)

Filters

Tool

☒ toolLong

☒ toolShort

Bugs

☐ My Bugs

☒ All Bugs

Vulnerability Type

☐ SQL Injection

☐ XSS

Example: RQ 1

■ Prototype 2

D SAT Interface

Project: Alpha

toolShort

Name (Bug title)	Type	Fix Location	Assignee
FI_EMPTY	FI	12.4 EditList.java	Varma
CO_SELF	CO	11.2 StringComparer.java	Un-assigned
DMI_EMPTY	DM	3.3 DatabaseHelper.java	Elina

toolLong

Name (Bug title)	Type	Fix Location	Assignee
FI_EMPTY	FI	12.4 EditList.java	Varma
EQ_CHECK	EQ	6.3 LoopHelper.java	Max
XSS_REQUEST	XSS	5.4 HttpSender.java	John

Filters

- ☐ My Bugs
- ☒ All Bugs

Marker Type

- ☐ Source
- ☒ Sink
- ☐ Fix Locations

Vulnerability Type

- ☐ SQL Injection
- ☒ XSS

Thesis Work Plan

- Problem Statement
- Research Questions
- What Current Tools do?
- Our Approaches
- Evaluation
- Time Plan

Evaluation

- Experimental Design

- Number of Test Users:

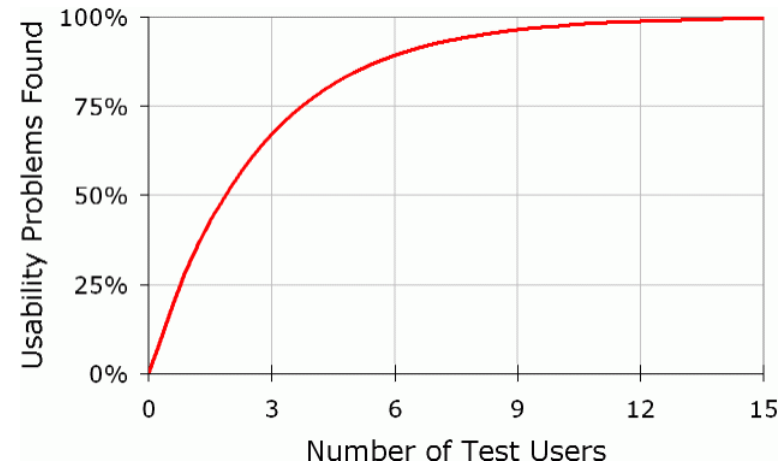
Dr. Nielsen recommends – **5**

Nielsen, Jakob, and Landauer, Thomas K.:

"A mathematical model of the finding of usability problems,"

- Order of evaluation:

Users tend to learn – order of presenting prototypes is altered



Evaluation – Usability Inspection Methods

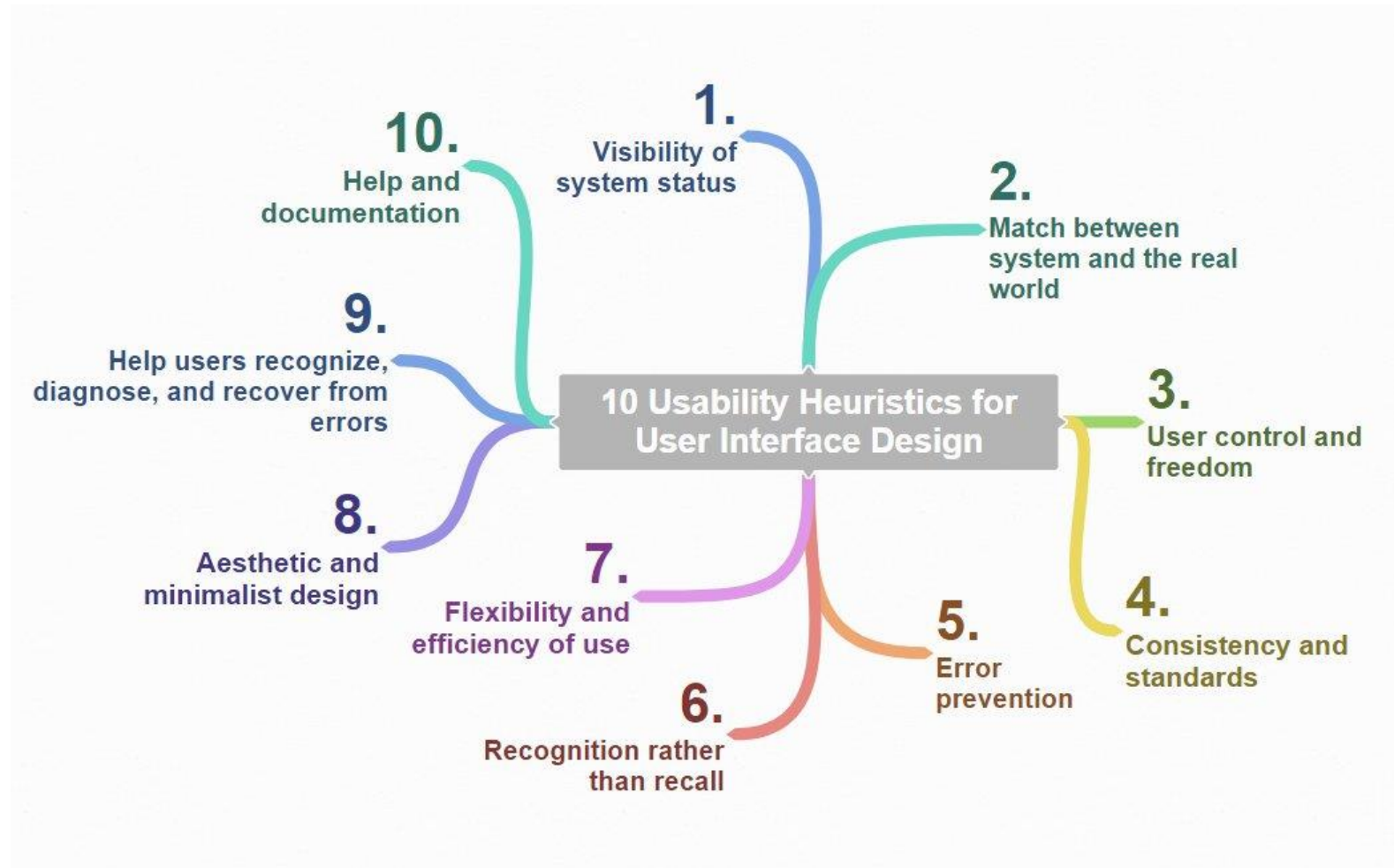
■ Cognitive Walkthrough

For each step to a predefined task, the following aspects are analysed.

- Will the user try and achieve the right outcome?
- Will the user notice that the correct action is available to them?
- Will the user associate the correct action with the outcome they expect to achieve?
- If the correct action is performed; will the user see that progress is being made towards their intended outcome?

Evaluation – Usability Inspection Methods

■ Heuristic Evaluation



Evaluation – Usability Inspection Methods

■ Heuristic Evaluation

Each problem w.r.t. a heuristic is rated accordingly; 0 – 4

0 - do not agree this is a usability problem

1 - cosmetic problem

2 - minor usability problem

3 - major usability problem (important to fix)

4 - usability catastrophe (imperative to fix)

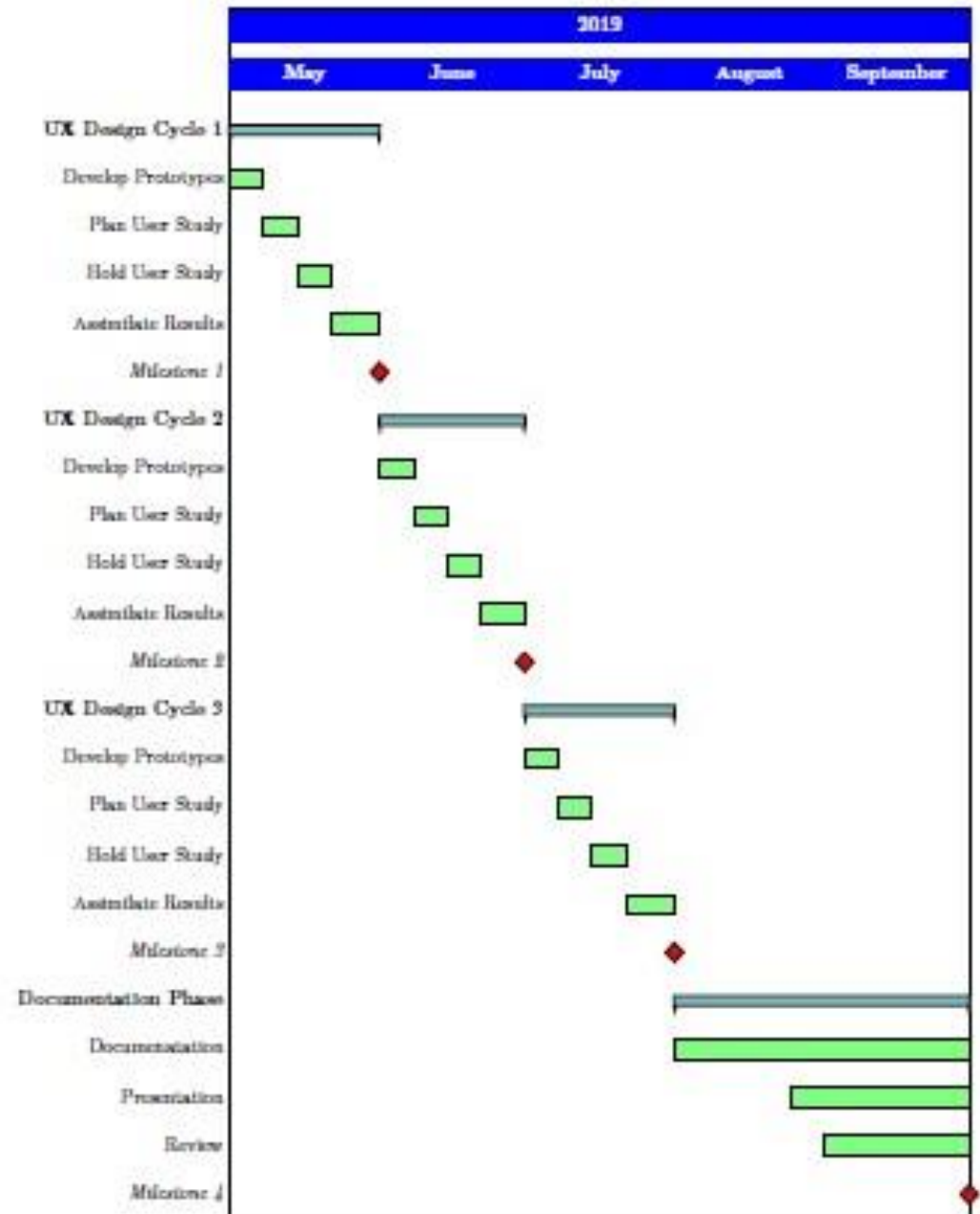
Comparative Study

Thesis Work Plan

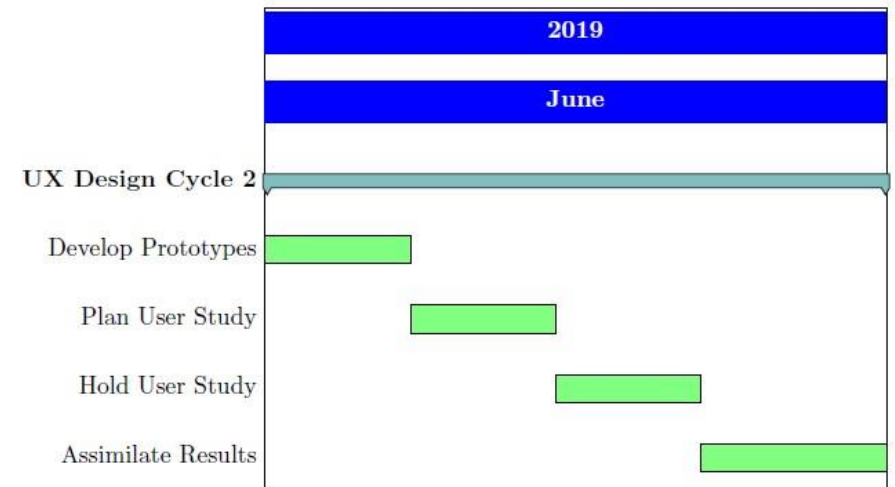
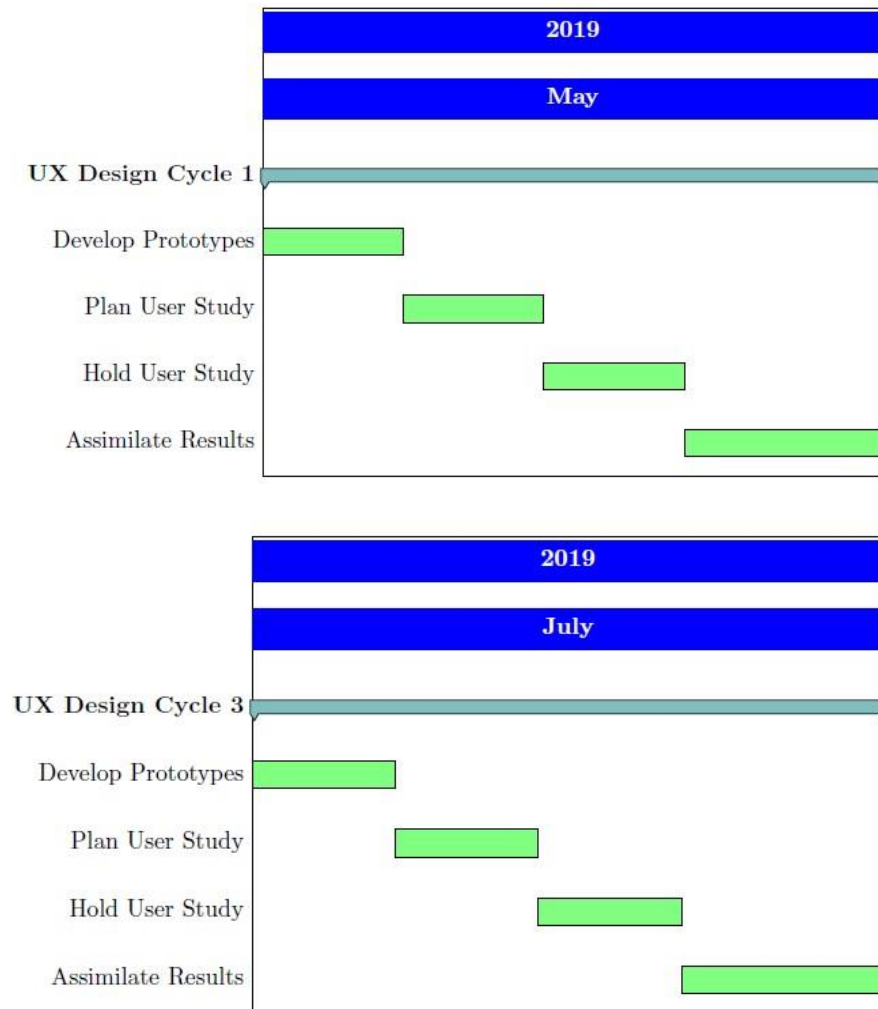
- Problem Statement
- Research Questions
- What Current Tools do?
- Our Approaches
- Evaluation
- Time Plan

Time Plan

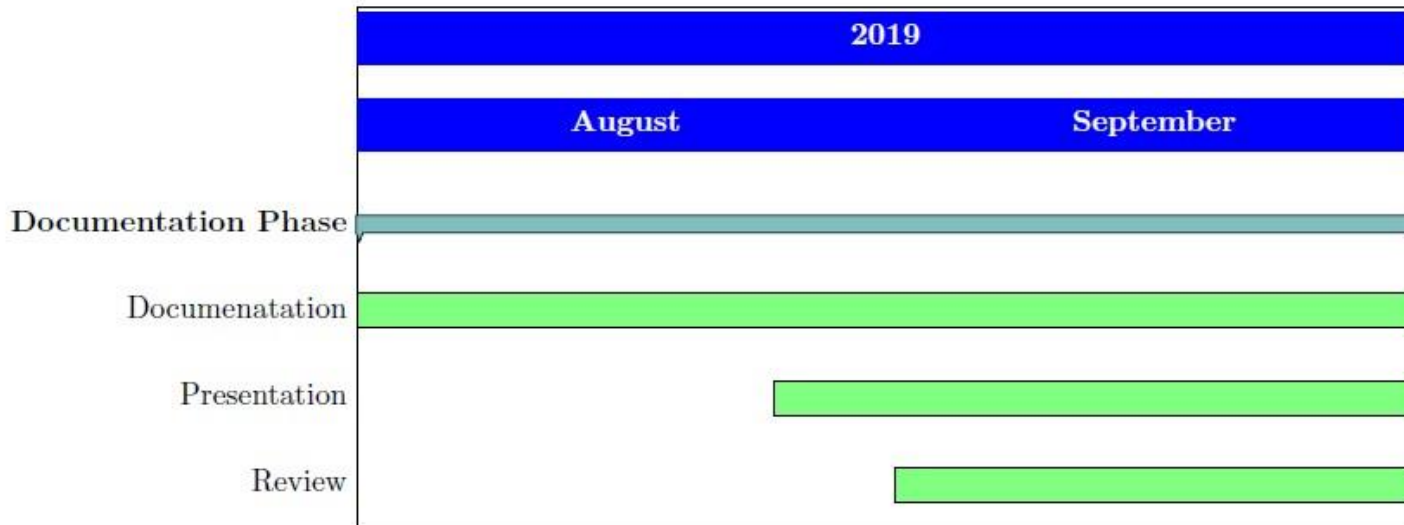
- Official Time: 5 Months
- Milestones: 4



Milestones 1 2 3



Milestone 4



References

- [1] *A Survey of Static Program Analysis Techniques*. url: <https://www.ics.uci.edu/~lopes/teaching/inf212W12/readings/Woegerer-progr-analysis.pdf>
- [2] *Balsamiq. Rapid, effective and fun wireframing software.* | *Balsamiq*. url: <https://balsamiq.com/>.
- [3] Olga Baysal, Reid Holmes, and Michael W. Godfrey. “No issue left behind: reducing information overload in issue tracking”. In: *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2014*. Ed. by Shing-Chi Cheung, Alessandro Orso, and Margaret-Anne Storey. New York, New York, USA: ACM Press, 2014, pp. 666–677. isbn: 9781450330565. doi: 10.1145/2635868.2635887.
- [4] Al Bessey et al. “A few billion lines of code later: using static analysis to find bugs in the real world”. In: *Communications of the ACM* 53.2 (2010), pp. 66–75.
- [5] Marilyn Hughes Blackmon et al. “Cognitive walkthrough for the web”. In: *Proceedings of the SIGCHI conference on human factors in computing systems*. ACM. 2002, pp. 463–470.
- [6] Lorraine Borman. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY: ACM, 1985. isbn: 0897911490. url: <http://dl.acm.org/citation.cfm?id=317456>.
- [7] *Checkmarx – Application Security Testing and Static Code Analysis*. url: <https://www.checkmarx.com/>.
- [8] Maria Christakis and Christian Bird. “What developers want and need from program analysis: an empirical study”. In: *Automated Software Engineering (ASE), 2016 31st IEEE/ACM International Conference*. IEEE. 2016, pp. 332–343.
- [9] John David Colleran, Gerardo Bermudez, and Vadim Gorokhovky. *Responsive user interface to manage a non-responsive application*. US Patent 6,850,257. Feb. 2005.
- [10] Aurelien Delaitre et al. “Evaluating Bug Finders–Test and Measurement of Static Code Analyzers”. In: *2015 IEEE/ACM 1st International Workshop on Complex Faults and Failures in Large Software Systems (COUFLESS)*. IEEE. 2015, pp. 14–20.
- [11] *Designing code analyses for Large Software Systems (DECA)*. url: <https://www.hni.uni-paderborn.de/swt/lehre/deca/>.
- [12] Alan Dix et al. “Spreadsheets as User Interfaces”. In: *Proceedings of the International Working Conference on Advanced Visual Interfaces - AVI '16*. Ed. by Maria Francesca Costabile et al. New York, New York, USA: ACM Press, 2016, pp. 192–195. isbn: 9781450341318. doi: 10.1145/2909132.2909271.
- [13] dustinca. *Proceedings of the 2nd Workshop on Refactoring Tools*. New York, NY: ACM, 2008. isbn: 9781605583396. url: <http://dl.acm.org/citation.cfm?id=1636642>.

References

- [14] *FindBugsTM - Find Bugs in Java Programs*. url: <http://findbugs.sourceforge.net/>.
- [15] *FindBugsTM - GUI Scan Results*. url: <http://findbugs.sourceforge.net/manual/gui.html>.
- [16] Lori Flynn et al. "Prioritizing alerts from multiple static analysis tools, using classification models". In: *Proceedings of the 1st international workshop on software qualities and their dependencies*. ACM. 2018, pp. 13–20.
- [17] *Gamification | Coursera*. url: <https://www.coursera.org/learn/gamification>.
- [18] Garima Gaur, Sumit Kalra, and Arnab Bhattacharya. "Patterns for Indexing Large Datasets". In: *Proceedings of the 23rd European Conference on Pattern Languages of Programs - EuroPLoP18*. Ed. by Unknown. New York, New York, USA: ACM Press, 2018, pp. 1–6. isbn: 9781450363877. doi: 10.1145/3282308.3282314.
- [19] Shinpei Hayashi et al. "Historef: A tool for edit history refactoring". In: *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*. IEEE, 2/03/2015 - 06/03/2015, pp. 469–473. isbn: 978-1-4799-8469-5. doi: 10.1109/SANER.2015.7081858.
- [20] Lars Heinemann, Benjamin Hummel, and Daniela Steidl. "Teamscale: Software quality control in real-time". In: *Companion Proceedings of the 36th International Conference on Software Engineering*. ACM. 2014, pp. 592–595.
- [21] James J Horning. "What the compiler should tell the user". In: *Compiler Construction*. Springer. 1974, pp. 525–548.
- [22] *How to Change Your Career from Graphic Design to UX Design*. url: <https://www.interaction-design.org/literature/article/how-to-change-your-career-fromgraphic-design-to-ux-design>.
- [23] Brittany Johnson et al. "Why don't software developers use static analysis tools to find bugs?" In: *Proceedings of the 2013 International Conference on Software Engineering*. IEEE Press. 2013, pp. 672–681.
- [24] Erica Mealy et al. "Improving Usability of Software Refactoring Tools". In: *2007 Australian Software Engineering Conference (ASWEC'07)*. IEEE, 10/04/2007 - 13/04/2007, pp. 307–318. isbn: 0-7695-2778-7. doi: 10.1109/ASWEC.2007.24.
- [25] Na Meng et al. "An approach to merge results of multiple static analysis tools (short paper)". In: *2008 The eighth international conference on quality software*. IEEE. 2008, pp. 169–174.

References

- [26] Lisa Nguyen Quang Do and Eric Bodden. “Gamifying Static Analysis”. In: *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ESEC/FSE 2018. Lake Buena Vista, FL, USA: ACM, 2018, pp. 714–718. isbn: 978-1-4503-5573-5. doi: 10.1145/3236024.3264830.
- [27] Jakob Nielsen. “Usability inspection methods”. In: *Conference companion on Human factors in computing systems*. ACM. 1994, pp. 413–414.
- [28] OASIS. url: <https://www.oasis-open.org/>.
- [29] OASIS SARIF TC. url: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=sarif.
- [30] OASIS SARIF TC: Repository for development of the draft standard. url: <https://github.com/oasis-tcs/sarif-spec>.
- [31] *Observe, Test, Iterate, and Learn (Don Norman) (Video)*. url: <https://www.nngroup.com/videos/observe-test-iterate-and-learn-don-norman/>.
- [32] Daniel Plakosh et al. *Improving the Automated Detection and Analysis of Secure Coding Violations*. Tech. rep. CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST, 2014.
- [33] *Response Time Limits: Article by Jakob Nielsen*. url: <https://www.nngroup.com/articles/response-times-3-important-limits/> (visited on).
- [34] *Sample Of Covered Software Vulnerabilities (OWASP Top 10 and more)*. url: <https://www.checkmarx.com/technology/vulnerability-coverage/>.
- [35] *SARIF Example*. url: <https://blogs.grammatech.com/static-analysis-results-a-format-and-a-protocol-sarif-sasp>.
- [36] *Software Fail Watch*. url: <https://www.tricentis.com/news/software-fail-watch-says-1-1-trillion-in-assets-affected-by-software-bugs-in-2016/>.
- [37] *SWAMP SCARF to SARIF*. url: <https://github.com/mirswamp/swamp-scarf-sarif>.
- [38] *Teamscale*. url: <https://www.cqse.eu/en/products/teamscale/features/>.
- [39] *The Definition of User Experience (UX)*. url: <https://www.nngroup.com/articles/definition-user-experience/>.
- [40] Christoph Treude, Ohad Barzilay, and Margaret-Anne Storey. “How do programmers ask and answer questions on the web?” In: *Proceeding of the 33rd international conference on Software engineering - ICSE ’11*. Ed. by Richard N. Taylor, Harald Gall, and Nenad Medvidovic. New York, New York, USA: ACM Press, 2011, p. 804. isbn: 9781450304450. doi: 10.1145/1985793.1985907.

References

- [41] *Usability 101: Introduction to Usability*. url: <https://www.nngroup.com/articles/usability-101-introduction-to-usability/>.
- [42] *Usability Engineering : Book by Jakob Nielsen*. url: <https://www.nngroup.com/books/usability-engineering/>.
- [43] Shaowei Wang, David Lo, and Lingxiao Jiang. “An empirical study on developer interactions in StackOverflow”. In: *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. ACM. 2013, pp. 1019–1024.
- [44] *Why You Only Need to Test with 5 Users*. url: <https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/>.
- [45] [Nielsen, Jakob, and Landauer, Thomas K.: "A mathematical model of the finding of usability problems."](#) *Proceedings of ACM INTERCHI'93 Conference* (Amsterdam, The Netherlands, 24-29 April 1993), pp. 206-213.

Thank you for listening...



Summary

- Importance of Static Analysis tools
- Usage of Multiple Static Analysis tools
- Future scope of SARIF

- Need for a single user interface for multiple tools
- It should be Usable

- This Thesis follows UX Design Cycle to achieve usable prototypes.