

# Improving Human-Centric Software Defect Evaluation, Reporting, and Fixing

Kenny Huynh, Juvent Benarivo, Chew Da Xuan, Giridhar Gopal Sharma,  
Jeffrey Kang, Anuradha Madugalla and John Grundy

*Department of Software Systems and Cybersecurity, Faculty of Information Technology  
Monash University, Melbourne, Australia*

{khuy22, jben0001, dxche5, gsha0009, jkan66}@student.monash.edu, {anu.madugalla, john.grundy}@monash.edu

**Abstract**—Defect reporting customarily exists in most applications and web sites to support issue reporting by end users and for developers to receive actionable feedback. However the impact of “human-centric” issues - such as age, gender, language, culture, physical and mental challenges, and socio-economic status - is often overlooked in the development process and during product inception and defect reporting. Most defect reporting tools lack necessary human-centric features to enable a challenged user to adequately navigate and report defects i.e. do not take into account the human differences between end users that cause defects for them in their software and make reporting of such defects difficult for them. Most defect reporting tools also lack sufficient defect report structuring, reporting guidance, and do not emphasize the possible perceived severity of the defect to developers from end users who are very different to them. If users are unable to report defects due to usability issues, this makes those same defect reports difficult to understand by the developer. In this paper, we aim to improve human-centric defect reporting by employing cognitive walkthrough with diverse end user personas, develop a prototype of an improved defect reporting tool, and evaluate the prototype’s defect reporting process from end user and developer perspectives. Our findings offer a foundation for improved human-centric defect evaluation, reporting and fixing.

**Index Terms**—human-centric software defects, cognitive walkthrough, disabilities, personas, defect reporting, bug reporting

## I. INTRODUCTION

In most modern day applications such as mobile, web, and even desktop there is a process to share user feedback to the developer in the form of reporting a software defect or bug. Such *software defects* are the deviations or irregularities between the actual and expected results of a software application [1], [2]. They manifest through software use and are incidentally and inherently generated during the development phase of a system as well as through perception of a user. These defects reflect an instance of frustration for a user and can arise in a multitude of forms, classified as, functional: arising as aspects of the interface or system as a whole and non-functional: produced via usability, performance, reliability, and human-centric issues.

*Human-centric design* in product development, is an approach that puts user needs, desires and abilities at the forefront of the development process [3]. It involves making educated and informed design decisions based on how people can/need/want to perform tasks, instead of expecting users to

adjust and accommodate their behaviors to fit the product. Most of these issues arise as a byproduct of certain human differences. For example, an individual with deteriorated motor control or mobility impairment may find applications with complex navigational mechanisms or excessive scrolling a human-centric defect, as it inadvertently causes the individual to refrain from using such software. Users have different languages, educational levels, culture, age and gender. All of these may mean one user encounters problems using the application that another does not [4]–[7]. As such, it is important for developers to be able to responsively resolve these types of issues.

In software development, bug reports provide crucial information to developers; defects can be more quickly and concisely resolved if the report is tailored in a way that is useful to the developer [1]. Furthermore, as previously studied, valuable defect reports should include accurate steps for defect reproduction, supporting attachments and evidence, observed behavior, and application configuration, to name a few [8]. However, this is often not the case: usability defect reports are often unclear and incomplete, and can widely differ by quality due to a range of factors [2], [8].

In reporting usability defects, users with difficulties often tend to share less useful information due to their lack of knowledge about defect reporting tools, processes and their lack of reporting experience [9]. This ultimately impacts the quality of defect reports. In a prior study on “what makes a good bug report”, researchers discovered an information mismatch between what developers need and what users report [2]. The information provided by users is usually without guidance and clarity and presents itself as a convoluted mix of information [10]. Additionally, defect reporting has many other issues such as rarely supplying usability-related information making it difficult for developers to determine the cause of the defect [11]; lack of potential suggestions about possible solutions for the defect [12]; and issues with textual coherence and the availability of stack traces [1].

These limitations, in conjunction with the human-centric defects formed through user perception, establish the driving force and motivation of our research. Based on review of existing literature on software and usability defect reporting, as well as reviewing the Web Content Accessibility Guidelines [13], we set out to improve the human-centric defect reporting

process. To do this we developed a set of personas to represent a range of end users with quite different human aspects (age, gender, physical and mental challenges, language etc). We then designed and prototyped a new set of defect reporting tools for web sites and mobile phones that try to take into account (i) the range of end user challenges encountered when using apps/web sites i.e. **human-centric defects** to report, and (ii) the range of end user challenges encountered when trying to **report these human-centric defects**. We also developed personas of developers that are quite different to the end users reporting these human-centric defects. Then we investigated how to present the reported defects to the developers in order to help them better understand the defect and to fix them.

From this study we developed a set of guidelines that can improve defect reporting for people with usability issues, such as vision, hearing, motor and reading impairments. These guidelines help to overcome possible interface and interaction issues which currently hinder these users ability to generate useful defect reports. Another significant outcome of this study is the identification of factors that can assist developers in human-centric defect evaluation and resolution. The first factor was that educating users about defect reporting as well as educating developers about personas of different possible users and their diverse challenges was useful. Second was that features such as capturing the frequency of the use of application/encountering a defect doesn't affect the developers perceived severity of the issue. The last key factor was that increasing the amount of extra information collected about a defect, while taking appropriate steps to prevent over complication of defect reports, was effective.

The rest of this paper is organised as follows. Section II explains our research questions. Section III details our methodology of creating personas and applying them on chosen applications. Based on our findings we carried out a human-centric design and prototyping of a new set of defect reporting tools. Section IV presents our evaluation, where we applied a cognitive walk-through on our prototype with the personas and identified strengths and limitations from this analysis. Section V summarises the limitations in our study and key future work that needs to be undertaken and Section VI summarises our review of key related work in this area.

## II. MOTIVATION

Most defect reporting tools, such as JIRA, BugZilla and those of GitHub, are designed for reporting conformance-type errors i.e. logical faults in the developed software. Yusop's work has demonstrated these tools poorly suit reporting usability defects [11]. Similarly, most software defect taxonomies focus on logical error defects. Most existing usability defect taxonomies have numerous limitations, including lacking support for modern smart-phone and tablet interface defects, lacking adequate indications of defect severity, and poorly supporting diverse end users in reporting defects they encounter relating to their personal human characteristics [14].

As an example, we can consider a scenario where a colour-blind user is attempting to use an app, but finds some serious

issues in usability due to poor choice of colours in text and images. They can then bring up the defect reporting form in the app to report this error to the app developers. But they may then find that the defect reporting form in the app itself poorly supports their colour-blind challenges: it has poor choice of colour meaning its very difficult to see and read text field and button labels; there is no way to magnify text, change colour choice, overlay filter, or report error via voice rather than typing text. When they do manage to generate a defect report and this is received by the development team, the developer assigned to investigate and fix the error can not understand nor even find the problem reported, as they are not colour-blind. Similar issues for differently aged users [5]; users with different language needs (spoken language or terminology used) [6]; those with various physical and cognitive challenges [7]; different gender [15]; and so on.

To address these issues, we developed a set of research questions, targeting improved human-centric defect reporting support for end users, and improved defect fixing support for developers. The first set of questions focuses on improving defect reporting by people with usability issues. The second set of questions discusses ways of assisting developers in human-centric defect evaluation and resolution. We classified these as "user-centric" and "developer-centric" issues:

### **How can we improve defect reporting for people with diverse usability issues?**

- 1) How to improve defect reporting for people who are visually impaired/hearing impaired/has issues in reading? (interface issues) [RQ1]
- 2) How to improve defect reporting for people who have trouble physically interacting with the application/software? (interaction issues) [RQ2]

### **How can we assist developers in better human-centric defect evaluation and resolution?**

- 1) Does educating users about defect reports improve a developer's understanding of the defect? [RQ3]
- 2) Does capturing the user's frequency of application use (and/or frequency of encountering the issue) affect or increase the perceived severity of the issue by a developer? [RQ4]
- 3) How does increasing the defect form's complexity/fields affect the developer? [RQ5]

## III. OUR APPROACH

Figure 1 shows an outline of our improved human-centric defect reporting process. (1) We developed a range of personas for example end users and developers. (2) We used these to design a set of defect reporting forms, following prior work on usability defect reporting tool improvements [11]. (3) We further refined these defect reporting interfaces to support users with various sight challenges and cognitive challenges. (4) We designed interfaces to present human-centric defect reports to developers along with persona(s) representing the different defect reporter key human characteristics. (5) We performed a number of cognitive walk-throughs of our prototype with

different end user personas, developer personas, and representative human-centric defects to report, evaluate and fix. (6) We fed back findings from the cognitive walk-throughs to refining our human-centric defect reporting prototype designs.

### A. Establishing Personas

Multiple approaches were considered to gather requirements for our improved defect reporting process from a human-centric view. Due to the current COVID-19 situation, it was hard to recruit volunteers and conduct usability analysis on defect reports. Therefore the team decided to conduct requirement identification using personas. We established personas using the following five human centric issues. We chose these as there are a good range of literature on impact on usability of these issues. Our approach does not preclude other human-centric differences e.g. differently aged users, different culture and language users, etc.

- 1) Colour blindness: [RQ1]
- 2) Dyslexia and Aphasia: [RQ1]
- 3) Hearing Impairment: [RQ1]
- 4) Mobility or Dexterity Impairment: [RQ2]
- 5) Vision impaired (Screen Reader User): [RQ1]

We assigned one of the above human-centric issues per persona and created five personas in total. For each persona we entered detailed background information about the issue and added goals that each persona wanted to achieve when reporting their assigned bugs. We informed our persona development from literature documenting issues commonly faced using web sites and apps by people with each of these challenges [16]–[18]. Below we present a summary of the five personas that we developed and used in this work.

**Colour Blind Persona:** The colour blind persona reflected a common issue faced by most people with colour blindness;

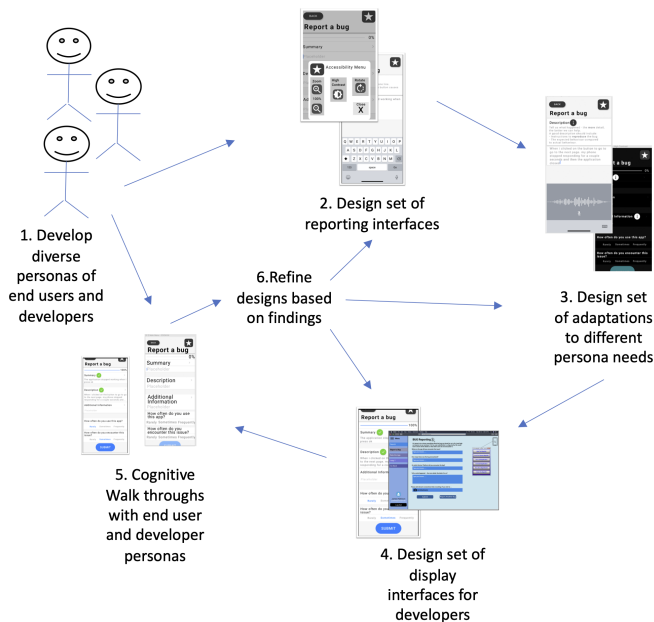


Fig. 1: Overview of our approach

the lack of ability to distinguish between certain colours. This specific Persona had Deuteranopia, otherwise known as red green colour blindness, as shown in Figure 2. This form of colour blindness causes people to identify shades of green colour in red shades [16]. There are other kinds of colourblindness that could be investigated using modified personas.

**Dyslexia and Aphasia Persona:** People with dyslexia usually have trouble with processing written language, which can deter them from detailed reading [18]. In addition to this, they may have some issues with spelling as well. Aphasia is an impairment of language, affecting the production or comprehension of speech and the ability to read or write. Additionally, these users might also have difficulties in expressing themselves when speaking or writing and they may also have issues with mixing up sounds or words in general [19]. To reflect these characteristics, the Dyslexia and Aphasia Persona presented a user that had trouble in processing both written and spoken language. The user particularly had poor spelling, difficulties in summarizing stories and remembering things like a PIN or telephone number.

**Hearing Impairment Persona:** There is a wider range of variation with how much the user can hear or distinguish audio stimuli [20]. A user could be hard of hearing with moderate hearing impairments in their ears or otherwise have a substantial diminished ability to hear. Individuals who are deaf, usually rely on caption or transcription to understand any audio or media content. Some can speak but they may choose not to because it is difficult for them to regulate the volume, pitch or sound they want. For some, sign language is the primary language, and they may not read the written language as fluently, hence they might rely on simpler text which is supplemented by images, graphs, or other illustrations. We incorporated most of these characteristics to our persona.


<i>Name</i>	ROBERT PATTINSON	
<i>Age</i>	32	
<i>Gender</i>	Male	
<i>Occupation</i>	Lecturer	
<i>Disability Information</i>	Has a common form of colour-blindness (Red-green colour blindness/ Deuteranomaly) which came from a past trauma (accident). This form of colour-blindness is common amongst males where the ability to identify the green colour is weaker making all green colour shades like red shades. In the image above, Robert cannot distinguish between the red and green dots failing to identify the number '74' in green. Robert identified the circle with different shades of green dots. In addition to this, some green colour visuals are seen in yellow colour by Robert.	
<i>Issues Encountered</i>	<ul style="list-style-type: none"> <li>Wants to be able to file a bug report based on similarly colored buttons.</li> <li>Wants a detailed alert button instead of a red button to avoid missing any alerts or warnings.</li> <li>Wants to avoid use of colors such as red and green in any UI design; or prefers a customized choice of colors in display to ease his preference of viewing, as he finds it difficult to distinguish between such colors.</li> </ul>	

Fig. 2: Persona created for a colour blind user

**Mobility or a Dexterity Impairment Persona:** There are multiple conditions that could cause mobility or dexterity related impairments such as arthritis, cerebral palsy, and essential tremor. Individuals with mobility or dexterity related impairments face issues that deal with extremity use and capabilities. These limit speed, strength, endurance, and coordination of limbs and cause difficulties in traversing and interacting with real world entities, web pages, and applications [21]. This means that these users will have difficulty with using controls like a mouse or keyboard and would prefer less mechanically intense tasks. Our Mobility or a Dexterity Impairment Persona was designed with these characteristics in mind.

**Vision Impairment Persona:** Vision impairment can have varying degrees of human vision. We designed this persona about a user who is completely blind and is using a screen reader tool to achieve UI navigation. Users with vision impairments also have the option of using braille readers with static or refreshable displays [17]. However, in this persona we are focusing on a user who prefers screen readers due to their wider availability.

### B. Cognitive Walk-through

Once the above personas were designed, the research team conducted a cognitive walk-through for each of them on a set of applications to evaluate usability issues. When available, both the desktop and mobile sites of these applications were used for the evaluation. The chosen applications were Grab, a university Moodle, Snapchat and Skype. The reasons for choosing these applications include being popular applications in 2020, their previous reputation as examples of bad user interface design, and their low star rating on the App store and Google Play store, especially reviews relating to accessibility and usability issues [22], [23]. The key steps included: a team member took a persona and web site or app; worked through the website or app to carry out a candidate task; identified a human-centric defect this persona would encounter using the website or mobile app; and used our web or mobile defect reporting prototype to “report” the defect. They recorded challenges they encountered describing the defect and using the reporting prototype.

The main outcome of this cognitive walk-through was a list of potential improvements that could be introduced in the bug reporting process for such apps and web sites.

For the **colour blind user**, the main findings from the cognitive walk-through proved findings from an earlier research which were, to use variations of contrasting colour schemes: allow users to visually identify UI components; use additional symbols: help to convey messages without relying only on colour [24].

For **Dyslexia and Aphasia user**, a key finding was to include a text to speech or voice recognition feature. This was an option to help users communicate in their preferred format and would often help clearer communication to the developer. Other key findings include: Using a spell checker tool; Using smaller and succinct paragraphs: Easier to understand than

larger ones; Using a Sans Serif font: to increase readability [25]; Avoiding italics or underlines: These make text harder to read instead using bold to highlight parts of the text; and Provide supporting icons or pictures: Allows users to associate meanings with images.

The user with **Deafness or general hearing impairment**, faced less issues since audio stimuli was not crucial in the defect reporting process. However, we were able to find additional issues that should be still be taken into consideration: Including captions and transcriptions for any speech or other important sound effects used; Avoid using complicated text: Some deaf users may also have trouble with reading written language.

For the user with **Mobility or a dexterity impairment**, most issues arose from convoluted and smaller screens, where users have difficulties in interacting with UI elements: Implementing scaling elements or screen magnification: Allows those with smaller resolutions to interact with elements that are larger than normal through scaling their size; Add button to allow screen orientation change : This will accommodate users who want to rotate their screen without physical means; Reduce scrolling: Lessen the stress for the users as they may have difficulty scrolling; Avoid implementing time sensitive functions: Some users will be slower than others and this would impact them negatively.

For the user with **Vision impairment**, who uses a screen reader we came up with following findings: Have an option to increase font size: To support those with low vision; Ensure that all elements on the application has the proper HTML element tags, so that screen readers can detect these elements and read out their label; Avoid making longer scrolling lists and put vital details near the top: Screen readers will read from top down and this will help users to grasp key information quickly; Do not force attachments or screenshots: The screenshots and attachments created by the user may not always have alt text tags, making it hard for the user to distinguish which attachments are relevant.

### C. Initial Prototype Development

Based on the information collected during the cognitive walk-through, we created prototype designs for a sample mobile and desktop application for defect reporting. Figure 3 shows an example of a desktop human-centric defect reporting design. Figure 4 shows examples from a set of mobile human-centric defect reporting designs. Due to time constraints we decided to progress only with the mobile application.

We created this mobile app human-centric defecting reporting prototype to explore how we could implement some of the suggestions and feedback gathered under each persona above. However, implementing all the requested functionalities would clutter the application, making it eventually very difficult to navigate. Therefore in the prototype, we tried to include as much functionalities as possible to satisfy the diverse user requirements while ensuring the defect reporting application itself maintained simplicity and clarity.

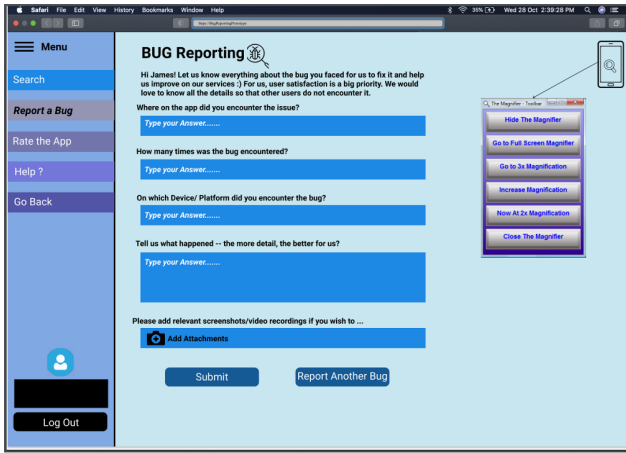


Fig. 3: Desktop application prototype for human centric defect reporting



Fig. 4: Mobile application prototype for human centric defect reporting

A defect report contains a few key features, such as a title, description, ways it can be repeated, suggestions for fixes, severity level, and areas for further attachments [11]. Our prototype human-centric defect reporting interface is made up from these key features since they potentially provide the most value to the developers. There is also an additional question to measure an individual’s frequency of use of the application that we are using to answer research question 2 of second set of questions. Figure 5a shows an example of the initial defect reporting form in the mobile app prototype.

We included a button to provide users with additional accessibility options. As highlighted earlier in section III-B, having the ability to zoom in, assists users who has trouble interacting physically with the screen. We also included a feature to change screen orientation via a button as an alternative to physically rotating the device. A high contrast mode was also included to provide extra visibility for users with vision issues.

On the report a bug page for summary, as shown in Figure 5c, there is an option to press an icon for additional information on what to write in the relevant sections. This was added to help answer research question 1 of set 2, where we try to educate the users on who should be writing specific bug reports. The additional information used in this section also had bolded words to assist users with issues in processing

written language.

Upon completing the relevant sections the users are presented a summary and this can be submitted with the click of a button as shown in Figure 5c.

Figure 6 shows examples of a user walking through the prototype defect reporting app to express a problem with an app they are using. They first specify a summary of the problem. They then specify further details of the problem they encountered. Note in the middle figure the user has decided not just to provide the developer textual feedback, but records a voice feedback as well. For some users it is much easier and faster to express their feedback via voice than typing text. The user can also optionally specify further information such as suggestions for fixing the problem, ways to repeat finding the problem, the kinds of difficulties it caused them, and so on.

In Figure 7, some examples of adapting the defect reporting tool to the end user’s preferences and challenges are shown for illustrative purposes. In the left hand example, a colour blind user has set the contrast and colour choices to make the display readable for them. In the middle example, a vision impaired user has asked for a larger, sans-serif font to be used. In the right hand example, a user has asked for the display to automatically resize and to guide them through the steps of reporting in a wizard-style approach, as proposed in [11].

When receiving a human-centric defect report the developer is presented with:

- the app in which the defect was reported
- one or more persona(s) representing the end user(s) reporting the defect
- a summary of the defect
- defect compulsory and optional details

The app and defect summary can be used to triage multiple defects but also redirect the defect to the most suitable developer to investigate and potentially fix it. The persona(s) are a general representation of one or more end users reporting the human-centric defect in the target app. The idea is that the personas will assist the developer to better understand the kinds of end user(s) reporting the defect; their particular challenges both in using the app and in reporting the defect; and in investigating the nature, severity and location of the defect in the app. Additional information might include suggestions for fixing the defect, or in replicating it using the provided persona(s) representing end users finding the defect.

#### IV. EVALUATION AND RESULTS

Once our new defect reporting prototype was completed, we conducted a second cognitive walk-through to evaluate it. Due to COVID-19 we were unable to evaluate our prototype directly with representative end users or developers. Our second cognitive walk-through instead consisted evaluating (i) the differently challenged end users ability to report a bug, and (ii) representative software developers ability to comprehend it and address it.

In the first part of our prototype evaluation we evaluated the prototype’s ability to support human-centric issue bug



(a) Prototype screen for defect report (b) Accessibility Menu for additional assistance (c) Report a bug page for summary - input (d) Completed Report a bug page

Fig. 5: Few chosen screens from the prototype

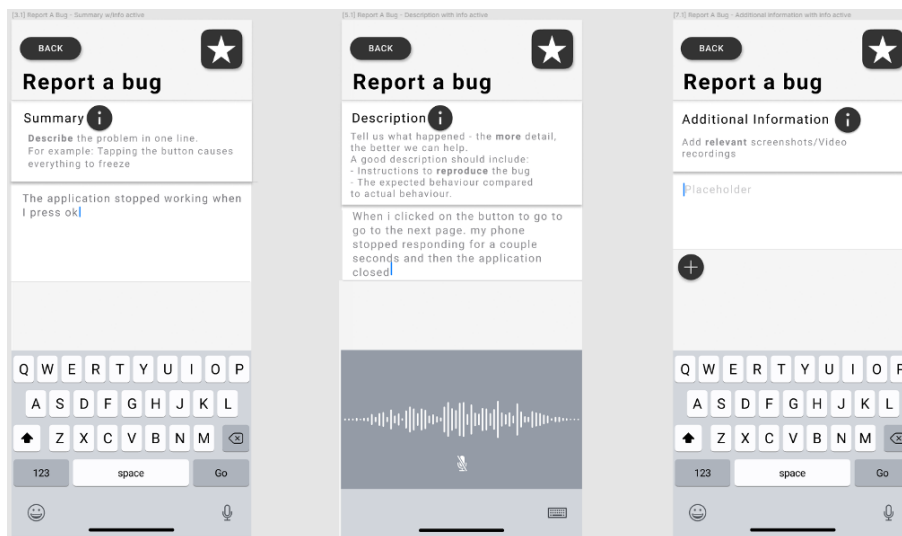


Fig. 6: Final prototype with information expanded

reporting by people with a variety of different disabilities without causing human-centric issues in the reporting itself. In this part each of our research team members re-enacted the persona that was described in Section III-A and reported a bug on our prototype. In the second part of the evaluation we evaluated the defect reports from the developer's perspective. This was done by each member of the research team using a different developer persona and evaluating the defect reports that would be sent in by the different personas. They then tried to identify the reported defects in the four evaluated apps

(Snapchat, Moodle, Skype and Grab) and what would need to be changed to fix these human-centric defects for each of the reporting end user personas.

The key goal of our research was to help people with different human-centric defects to report bugs they find in their apps without encountering issue in the bug reporting itself. In accordance with that goal, we tried to answer the research questions that were presented in Section II. For that purpose we gathered requirements, developed a prototype and conducted an evaluation. Through this process we found



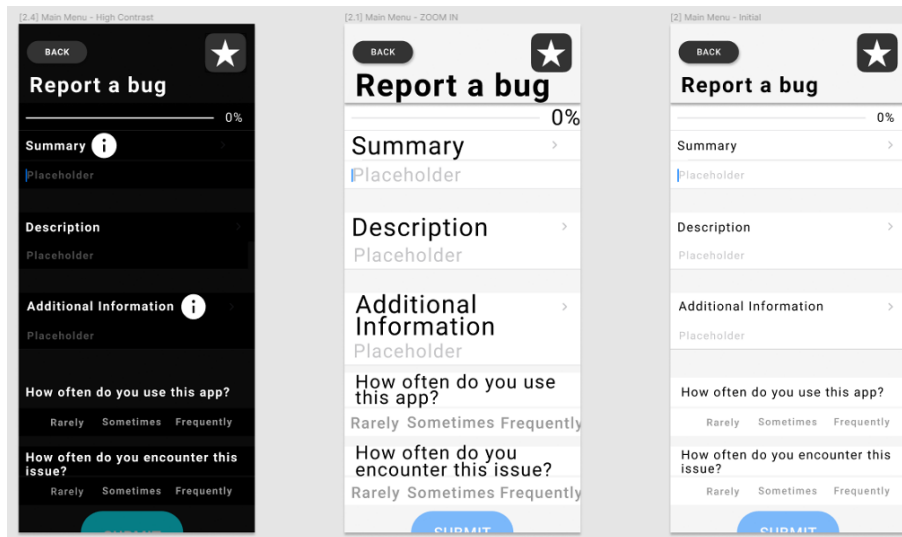


Fig. 7: Final prototype with Additional accessibility controls turned on

answers to our research questions and these are discussed below.

A. How can we improve defect reporting for people with usability issues?

**[RQ1]** How can we improve defect reporting for people who are visually impaired? (interface issues): Visually impaired users can be separated into two categories: users who have trouble reading and users who prefer to use screen readers. For users having *trouble reading*, the following key points must be kept in mind while creating defect reports around human-centric design. Some of these points such as “add contrast to separate text from background” is stated in the WCAG guidelines to improve readability as well.

- Include icons and images alongside words to help communication
- Break down content into shorter paragraphs if possible
- Use a Sans Serif font for any text (Arial, Comic Sans, etc.)
- The most used font size, is around 12 to 14 point, and should aim to be consistent
- Add contrast to separate text from the background
- Aligning content to the left should be a first preference as centre justify creates spaces in between texts
- Important keywords should be made bold
- Figure of speech must be in active voice as much as possible
- Avoiding any abbreviations of words that may confuse the reader

For users that adopt *screen readers*, the following key points must be kept in mind while creating defect reports around human-centric design:

- Avoid making long lists that require scrolling
- Put important text at the top of the screen, so that when users scroll through the webpage/ application it is easier for the user to see important text first

- Have an accessibility setting to increase font size
- Ensure all elements (icons, headers) have descriptions and labels using HTML 'alt text', 'longdesc' or 'ARIA' tags
- Do not force the users for any attachments such as screenshots, instead make it optional

Evaluation of our prototype from both of these perspectives indicated that it generally meets these requirements for these classes of end user. We did determine some language usage, complexity of language, lack of alternate text in some places, and other updates that were made to address some issues.

**[RQ2]** How can we improve defect reporting for people who have trouble physically interacting with the application/software? (interaction issues)

For users having issues in interacting with the application, the following points must be considered by defect reporting application creators:

- Implement additional navigational aids to assist the user with traversing the application
- Provide keyboard shortcut options that can be customised as per user need
- Avoid excessive amounts of typing and scrolling
- Add a button that can change orientation of the screen as part of easier accessibility

In general, our human-centric defect reporting prototype was able to provide all five of our personas with an accessible interface. This included allowing for voice recorded defect reports, colour; font and font size adaptation; zooming and rotation. As we evaluated the prototype with a cognitive walkthrough using personas, it is possible combinations of challenges are not sufficiently addressed. Similarly, as we simulated users based on the personas, we may have missed or mis-interpreted some particular challenges in both describing app human-centric defects, and in using the defect reporting app prototype.

## *B. How can we assist developers in human-centric defect evaluation and resolution?*

Our second set of cognitive walkthroughs helped us evaluate the bug reports captured from the first walkthrough above from the perspective of a developer. This was done by each member of the research team evaluating the defect reports sent in by the different personas we had created, using a different developer persona. The team members were of an age range between 18-25 and were final year IT undergraduates.

**[RQ3]** *Does educating users about defect reports improve a developer's understanding of the defect?*

By providing users with a guided defect report and additional information on what to include in a detail report, developers will usually obtain more information to help reproduce and resolve the issue.

Our defect reporting forms include sections for how to reproduce the bug and expected vs actual results. By providing these sections for the user, there will usually be more detail provided by the reporting user. This then helps developers to understand the defect, the context it occurred and which users and impacted. We included the persona reporting a defect to developers. This helped the to better understand the different end user challenges and thus reasons why the defect occurred. We found this particularly important when the developer does not share any of the human-centric characteristics of the end user(s) reporting the defect.

**[RQ4]** *Does capturing the user's frequency of use of the application (and/or frequency of encountering the issue) affect or increase the perceived severity of the issue by a developer?*

The frequency of using an application by the user gives the developer an idea about whether the application serves its purpose. A heatmap might be used to tell the developer which features of the application are most used/ which pages of the application are most visited by the user. This information can give a clearer idea as to whether the bug faced by user at a particular page/ while using a feature is a frequently faced bug or not.

However, the frequent use of an application may not communicate valuable information to the developer. Frequency of different bugs do not always correlate to the severity of the issue as multiple smaller bugs might not be prioritised over a larger security bug. Frequency of issues could give a general idea to the developer to fix the issues that are more frequently faced in a way helping developers prioritise tasks.

Additionally, the frequency of bugs on a particular component could also highlight additional issues on items related to that component.

**[RQ5]** *How effective is increasing the defect form's complexity/fields for the developer?*

Complexity of defect report forms are not directly equal to more information. The defect reports that were produced from our prototype were not overly complex, and were able to be easily understood by the developer. However as with most issues, the more information we can gather about a defect from the reporting end user, the faster and easier it is likely be resolved. The more details a user can provide the developer

about a bug faced by them, the clearer picture a developer gets to fix current issues and make user experience of high efficiency.

Furthermore, additional fields can be introduced to help the developer categorise and resolve issues. Modern issue tracking software, enables the customisation of these reports, adding further value and information attached on relevant issues.

There is however a point of diminishing returns, as too many fields make it harder for developers to troubleshoot the issue. If presenting the developer with a multitude of different resources and information on a single bug, it may not be entirely clear to the developer which section the bug originated from, as we may be attaching information that is widely irrelevant to the actual defect itself. Therefore a balance has to be ensured between not providing too much information or too less information. Our prototype evaluation found the level of detail about reported human-centric defects was reasonable for understanding most reported defects.

## V. DISCUSSION

Overall our prototype human-centric defect reporting tool evaluation proved to be very positive. We managed to develop a mobile app based defect reporting tool prototype that addressed issues of supporting end users with diverse human differences and challenges to (i) express a number of different human-centric defects in four common apps (Grab, Moodle, SnapChat and Skype), and (ii) report these defects to developers using a more human-centric defect reporting app. Our cognitive walk through with these defect reports and developer personas illustrated clear improvements in developers ability to understand and act on these human-centric defects reported as well as in users ability to report defects.

The first limitation is in the reliability of user "evaluation". During the user review, we decided to utilize personas to model end users with accessibility issues, instead of working with disabled participants. Additionally we used our team members to act as the developers who receive the defect reports. These were mainly due to the ongoing COVID-19 pandemic in 2020 as well as difficulties we faced in time constraints to obtain a suitable high-risk ethics approval for such a study. Therefore, this research has the potential to be further improved by conducting studies with actual disabled end users and actual developers. It would also be recommended to ensure such participants would cover different categories of society such as different ages, tech expertise, different job roles, etc. This would help to increase the breadth of the research and to mitigate any potential bias.

The second limitation is in way we utilized our personas. The primary purpose of utilizing a persona is to generate a focused and realistic representation of a particular category of potential end users, backed by adequate and consolidated research. If personas are to be continually used, it would be prudent to thoroughly detail them, as well as to employ the characteristics of the persona wholly, instead of merely considering it as an entity with a disability. During the cognitive walk-through, we only considered the characteristics



of the personas disability and ignored the other potentially important characteristics of these personas such as gender, age, or occupation. Therefore in a future research, it is recommended to apply the personas more thoroughly, to produce more insightful results from different perspectives.

Another limitation is our degree of coverage of the target end users for the four chosen apps to evaluate. Our defect reporting app prototype's design was based on the collated ideas from the review of the five personas in section III-A, and we understand that there is further room for improvement and human-centric issue coverage. The current prototype might benefit users possessing our five human-centric issues but there are many more human-centric issues that we have not considered, such as anxiety and dyscalculia, to name a couple. However adding additional features to address all additional disabilities can cause disadvantages for many other features. Therefore, our prototype was designed for general use by all users and any improvement or new features for a particular target user should be considered in future research via use of more adaptive interfaces.

By selecting a predetermined set of four common applications to perform cognitive walk-through with the use of personas, we have inadvertently generated a form of bias, and thus a validity threat to our research. As we, the researchers, have reviewed a predetermined set of applications that we have personally selected, there is a risk of confirmation bias, and ultimately an influence on the outcome of our evaluation. To minimize this bias, future research can be conducted with a random or standardized set of applications so that evaluators will not have prior impressions on specific applications.

A possible future work would be to try and determine any statistical significance to the finding that 'adding more defect report attributes has a positive effect on a developer's understanding of the defect'. Such a research with use of empirical analysis and qualitative insights can prove the existence of correlation or pattern between the number or addition of defect report fields and a developer's perception of the issue. In addition, such a study may also raise some observable quantitative discoveries such as an increase in defect resolution speed, which would empirically prove its use and become important in future defect report designs.

We want to extend our research, by exploring a wider range of potential human-centric solutions in order to tackle accessibility and usability issues that users may encounter. In terms of the UI experience, creating a design with a more sophisticated and elaborate design would contribute to improving the defect reporting process for a range of human-centric defects. The coherence of the process and of each defect report attribute is also important to consider and explore further in-depth. The embodiment of, for instance, intermittent suggestions or auto-fill throughout the defect reporting process may aid in these findings. Furthermore, certain features may also prove to be useful for users to report software defects, such as implementing other forms of media to convey information in consideration for those who may have issues with processing written language, or those who may prefer or be more visually

oriented, for example to inform the user of what makes the response of a particular report attribute useful for developers. Exploring the implementation of drop down selection for common fields or specialized fields would help to affirm our choice of not including this feature due to the consideration of other accessibility issues. In conjunction with exploring the effect of the amount of defect attributes, thoroughly exploring the possible uses of Likert scale attributes would help to affirm its practicality.

Finally, it may also be prudent to explore the usage of a desktop or web application and its implications towards improving human-centric software defect evaluation, reporting, and fixing. By employing these platforms, one could design and implement a classification or framework for keyboard shortcuts, that consider specific assistive technologies and/or certain keyboard permutations. The exploration and testing of accessibility of assistive technologies such as braille displays, or alternative pointing devices could also result in interesting findings with high utility in the field [26].

## VI. RELATED WORK

Software defect reporting has garnered much attention over the years in trying to improve the quality of software defect reports [8], [27]. However, there is a lack of attention on the human aspect of these defects [2]. Addressing the human-centric software defects are important to ensure universal access to software, especially for user groups such as people with motor issues, color blindness, agnosia and dyslexia.

A systematic literature review conducted to investigate the state of reporting usability defects from 2000 to 2016, shows numerous problems in usability defect reporting. One such is about "many reports being unsure about what information needs to be captured or how to capture them in usability defect reports" [2]. These researchers have developed a reporting tool which may solve one of the issues. However, more research such as these need to be conducted to understand the usability defects and to come up with solutions to report, evaluate and fix them.

Most defect reporting tools aim to retrieve maximum amount of information from the user but pay limited or no attention to user differences. At times this leads to huge chunks of information with lot of them being unimportant for defect identification by developers. This was shown by a study that aimed to create a guided defect report by encouraging users to fill in detailed information by splitting the fields into categories [11]. In spite of the provided guidance, the authors still found the collected information to be inadequate and the cause was attributed to lack of features to accommodate the handicapped.

Various studies have investigated issues in using apps and web sites by users with various challenges and human differences. This includes aging users [5], users with hearing and vision challenges [16], [17], [20], users with cognitive challenges [18], and users who use different languages, are of different genders, and come from different cultural backgrounds [6]. Little work has been done on how to report defects when using apps for users from these groups [15].

In order to support the developers in understanding the delivered information, some research also consider on bridging the gap between end users and developers. A further study [28] analysed open-source communities experience on using existing open-source defect reporting tools. The result revealed that generic defect forms are not suitable for usability defect reports. In this study, the open-source communities also suggested several improvements that could be made to enhance the experience of developers in resolving these defect reports. Another study conducted with software developers and usability defect reporters in order to identify what makes usability defects difficult to report, suggested that developers may not understand the significance of a usability defect. Therefore, this would understandably cause usability defects to receive less priority than functional defects [2].

## VII. SUMMARY

Our research involved creation of personas to represent end users with specific disabilities and differences that manifest human-centric issues when using apps and websites. We conducted a cognitive walk-through on a mix of mobile and web applications based on our created personas. We identified a number of design decisions to assist diverse end users report different challenges they have – human-centric defects – when using these apps. We then developed a prototype human-centric defect reporting app and carried out a second cognitive walk-through with our prototype to evaluate its performance. Our research questions and methods consisted of finding ways to assist both bug reporters and developers in reporting and understanding human-centric defects in apps and web sites. Our evaluation helped us to develop some guidelines to increase the usability of defect reporting tools for disabled end users and increase useful information about these defects to be provided to developers.

## ACKNOWLEDGMENT

Grundy is supported by ARC Laureate Fellowship FL190100035, and Madugalla is supported by ARC Transformation Hub IH170100013.

## REFERENCES

- [1] J. D. Strate and P. A. Laplante, "A literature review of research in software defect reporting," *IEEE Transactions on Reliability*, vol. 62, no. 2, pp. 444–454, 2013.
- [2] N. S. M. Yusop, J. Grundy, and R. Vasa, "Reporting usability defects: a systematic literature review," *IEEE Transactions on Software Engineering*, vol. 43, no. 9, pp. 848–867, 2016.
- [3] T. Winograd and D. Woods, "The challenge of human-centered design," *Human-centered systems: information, interactivity, and intelligence*, pp. 17–19, 1997.
- [4] J. Grundy, H. Khalajzadeh, and J. McIntosh, "Towards human-centric model-driven software engineering," in *ENASE*, 2020, pp. 229–238.
- [5] H. Hwangbo, S. H. Yoon, B. S. Jin, Y. S. Han, and Y. G. Ji, "A study of pointing performance of elderly users on smartphones," *International Journal of Human-Computer Interaction*, vol. 29, no. 9, pp. 604–618, 2013.
- [6] W. Smith, G. Wadley, O. Daly, M. Webb, J. Hughson, J. Hajek, A. Parker, R. Woodward-Kron, and D. Story, "Designing an app for pregnancy care for a culturally and linguistically diverse community," in *Proceedings of the 29th Australian Conference on Computer-Human Interaction*, 2017, pp. 337–346.

- [7] R. K. Bhardwaj and S. Kumar, "A comprehensive digital environment for visually impaired students: user's perspectives," *Library Hi Tech*, 2017.
- [8] N. Bettenburg, S. Just, A. Schröter, C. Weiss, R. Premraj, and T. Zimmermann, "What makes a good bug report?" in *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering*, 2008, pp. 308–318.
- [9] A. J. Ko and P. K. Chilana, "How power users help and hinder open bug reporting," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2010, pp. 1665–1674.
- [10] S. Davies and M. Roper, "What's in a bug report?" in *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 2014, pp. 1–10.
- [11] N. S. M. Yusop, J. Grundy, J.-G. Schneider, and R. Vasa, "Preliminary evaluation of a guided usability defect report form," in *2018 25th Australasian Software Engineering Conference (ASWEC)*. IEEE, 2018, pp. 81–90.
- [12] K. Hornbæk and E. Frøkjær, "What kinds of usability-problem description are useful to developers?" in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 50, no. 24. SAGE Publications Sage CA: Los Angeles, CA, 2006, pp. 2523–2527.
- [13] "Web content accessibility guidelines (wcag) overview — web accessibility initiative (wai) — w3c," <https://www.w3.org/WAI/standards-guidelines/wcag/>, (Accessed on 12/13/2020).
- [14] N. S. M. Yusop, J.-G. Schneider, J. Grundy, and R. Vasa, "A revised open source usability defect classification taxonomy," *Information and software technology*, vol. 128, p. 106396, 2020.
- [15] M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, and W. Jernigan, "Gendermag: A method for evaluating software's gender inclusiveness," *Interacting with Computers*, vol. 28, no. 6, pp. 760–787, 2016.
- [16] L. Jefferson and R. Harvey, "Accommodating color blind computer users," in *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility*, 2006, pp. 40–47.
- [17] A. Madugalla, K. Marriott, S. Marinai, S. Capobianco, and C. Goncu, "Creating accessible online floor plans for visually impaired readers," *ACM Transactions on Accessible Computing (TACCESS)*, vol. 13, no. 4, pp. 1–37, 2020.
- [18] M. J. Ransby and H. Lee Swanson, "Reading comprehension skills of young adults with childhood diagnoses of dyslexia," *Journal of learning disabilities*, vol. 36, no. 6, pp. 538–555, 2003.
- [19] H. Goodglass, *Understanding aphasia*. Academic Press, 1993.
- [20] W. Xiong, T. Yao, Q. Pan, and Z. Liu, "Usability heuristic evaluation for the hearing impaired language training mobile app," in *International Conference on Human-Computer Interaction*. Springer, 2020, pp. 294–307.
- [21] "Mobility impairments — disability resources & educational services - university of illinois," <https://www.disability.illinois.edu/instructor-information/disability-specific-instructional-strategies/mobility-impairments>, (Accessed on 01/28/2021).
- [22] T. Armano, M. Borsero, A. Capietto, N. Murru, A. Panzarea, and A. Ruighi, "On the accessibility of moodle 2 by visually impaired users, with a focus on mathematical content," *Universal Access in the Information Society*, vol. 17, no. 4, pp. 865–874, 2018.
- [23] D. Franzmann, L. Fischer, and R. Holten, "The influence of design updates on users: the case of snapchat," in *Proceedings of the 52nd Hawaii International Conference on System Sciences*, 2019.
- [24] W. Q. Khan, R. Q. Khan, M. Sarim, A. B. Shaikh, and S. K. Raffat, "An assistive model for ict applications for color blindness," *Research Journal of Recent Sciences ISSN*, vol. 2277, p. 2502.
- [25] F. T. Yoliando, "A comparative study of dyslexia style guides in improving readability for people with dyslexia," in *International Conference of Innovation in Media and Visual Design (IMDES 2020)*. Atlantis Press, 2020, pp. 32–37.
- [26] A. Kavcic, "Software accessibility: Recommendations and guidelines," in *EUROCON 2005-The International Conference on "Computer as a Tool"*, vol. 2. IEEE, 2005, pp. 1024–1027.
- [27] T. Menzies and A. Marcus, "Automated severity assessment of software defect reports," in *2008 IEEE International Conference on Software Maintenance*. IEEE, 2008, pp. 346–355.
- [28] N. S. M. Yusop, J. Grundy, and R. Vasa, "Reporting usability defects: limitations of open source defect repositories and suggestions for improvement," pp. 38–43, 2015.