CURRENT TRENDS IN INFORMATION COMMUNICATION TECHNOLOGY RESEARCH

ISSN: 2814-1709

CTICTR 2(1): 13 - 27 (June 2023)

Received:00-04-2023

Accepted:00-06-2023

A SHIFT LEFT-BASED ACCESSIBILITY MODEL FOR SOFTWARE DEVELOPMENT PROCESS IMPROVEMENT

Tolulope A. Awoniyi, MSc

Sunday A. Idowu, PhD

awoniyia@babcock.edu.ng
Department of Software Engineering
Babcock University
Ilishan-Remo, Ogun State, Nigeria
+234-907-775-2148

Shift Left-based Accessibility Model for Software Development Process Improvement

Tolulope Awoniyi [a]*, Sunday Idowu [b]

[a]*Department of Software Engineering Babcock University, Ilishan-Remo, Ogun State, Nigeria*

[b]*Department of Software Engineering Babcock University, Ilishan-Remo, Ogun State, Nigeria*

[a]*awoniyia@babcock.edu.ng*

[b]*Idowus@babcock.edu.ng*

Abstract

Accessibility modifications are required for individuals with situational, temporary, or permanent impairments to properly interact with web applications. Incorporating accessibility guidelines, such as the Web Content Accessibility Guidelines (WCAG), into software development, along with the software development life cycle and other paradigms, poses challenges. The linear progression of steps in a conventional software development process leads to the identification of problems at a later stage because testing is typically conducted toward the end. Limited attention to accessibility until the final development phase results in prolonged efforts, increased resource allocation, and higher expenses to address accessibility issues. Consequently, numerous software applications remain inaccessible to users with impairments. This study focuses on a Shift Left-based model to improve accessibility in the software development process. Employing the Shift Left a11y Model that emphasizes early integration for accessibility and coupling it with the Extreme Programming Development Life Cycle would enhance the integration of various testing practices and facilitate the development of software that is accessible to all users, including those with impairments.

Keywords: Accessibility, accessible software, shift-left, model, methodology for accessible software

1. Introduction

Many people have turned to the Internet as their primary source of information in recent years. Businesses are conducted online. Governmental services are offered on the Internet in several nations[1]. Currently, more than 1 billion people, or 16% of the world's population, experience some form of disability, and this number is rising, partly because of an aging population and a rise in the frequency of noncommunicable diseases [2]. As noted by [3] disability is an evolving concept and is the result of how people with impairments interact with environmental and attitudinal barriers that prevent them from fully and equally participating in society.

Web accessibility is a concern in web development. It is necessary to clarify exactly what web accessibility means. According to [4], web accessibility means that websites, tools, and technology are designed and developed with individuals with impairments in mind. People can perceive, understand, navigate, and interact with the Web and contribute to it. Web accessibility includes all forms of impairment that affect web access, such as cognitive,

auditory, physical, speech, neurological, and visual impairments. The web must be accessible to provide people with different disabilities with equal access and opportunities. Access to information and communication technologies, including the Internet, is recognized as a fundamental human right under the United Nations Convention on the Rights of Persons with Disabilities (UN CRPD) [4]. Web technologies can now address accessibility barriers in print, audio, and visual media considerably more easily. In certain circumstances, the Internet enables people with impairments to accomplish tasks that would otherwise be impossible. Accessible web applications can help people with impairments complete everyday tasks, increasing their participation in educational, sociocultural, and economic activities [5]. As a result, some organizations have proposed guidelines for developing accessible websites, such as the Web Content Accessibility Guidelines (WCAG) [6], and Section 508 of the Rehabilitation Act of 1973 [7].

Inaccessible web applications have caused barriers for individuals with impairments. 4.7% of the 174 countries' government health websites surveyed had fully adopted the Web Accessibility Initiative (WAI) accessibility guidelines [8]. The majority of countries' websites continue to have accessibility errors that create substantial impediments that prevent people with impairments from having equal access to and benefiting from essential health information during a global health crisis (the Coronavirus Disease 2019 also known as COVID-19 pandemic).

It can be particularly challenging to identify product errors in software development so that they can be addressed. To identify errors, software testing uses a wide variety of techniques. For success, the multiple types of software testing should be conducted and, more importantly, integrated into every stage of the software process. A project can reduce the number of errors and improve the quality of the code by testing frequently and early. The sequential nature of the processes in a traditional software development process causes problems to be discovered at a later stage. This helps explain why the Shift Left method focuses on finding and preventing errors from the outset of software development [3]. This study focuses on a shift left-based accessibility model for software development process improvement.

Incorporating accessibility guidelines such as WCAG into software development, along with the software development life cycle and other paradigms, poses challenges. The linear progression of steps in a conventional software development process leads to the identification of problems at a later stage because testing is typically conducted toward the end. Limited attention to accessibility until the final development phase results in prolonged efforts, increased resource allocation, and higher expenses to address accessibility issues. Consequently, numerous software applications remain inaccessible to users with impairments. Existing models consider accessibility in the development process. They primarily focus on integrating accessibility testing into each stage of software development limitedly, with stakeholder involvement not adequately emphasized. Thus, there is a need for the development of a model to incorporate accessibility early and throughout the software development process while enhancing testing practices and fostering effective stakeholder collaborations. Therefore, this study aims to design a shift-left-based accessibility model to improve accessibility in the software development process.

2. Literature review

Software engineering is essential in the development of accessible applications since it supports the integration between methodologies and specific accessibility techniques in the software development process [9], [10] maintains that the Agile model serves as a collection of software engineering best practices and principles. The Extreme Programming (XP) methodology is an agile software development process that aims to produce software of high quality and maximize the efficiency of the development team. In terms of the best engineering practices for software development, XP is one of the most popularly used agile models, especially for small-scale projects [9]. The merits of XP include high quality, continuous testing that results in a stable release with a small bug rate; low cost; rapid change allowed at any stage of the software development life cycle; continuous feedback from the project owner; and seamless code integration [11]. Shift Left is a method that focuses on finding and preventing errors from the outset of software development. It derives from the methodology view, where everything begins on the left side and ends on the right with deliveries or results. The sequential nature of the processes in a traditional software development process causes problems to be discovered at a later stage.

2.1. Accessibility Standards and Guidelines

The W3C is an international, independent body that develops web standards and protocols. The WAI and its W3C working groups establish accessibility guidelines for web browsers, web content, authoring tools, and evaluation tools. The WCAG was- created by the Accessibility Guidelines Working Group. The current standard is WCAG 2.1 [12]. Relevant standards similar to WCAG such as Section 508 of the Rehabilitation Act of 1973 [7]; the Information Technology-User Interface Accessibility (International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 29138-1:2018 [13]; the Guidance on Software Accessibility (ISO 9241-171:2008) [14]. WCAG and other accessibility guidelines and standards are discussed in this section.

2.2. The Extreme programming methodology

Extreme programming (XP) is an agile software development process that produces software of high quality and maximize the efficiency of the development team. Agile approaches are used by the great majority of development teams and projects [15]. XP was released by Kent Beck in 1996 [9]. In terms of the best engineering practices for software development, XP is the most specific agile framework. It was developed to improve software quality and provide an immediate response to a change in customers' requirements by breaking down the software development process into manageable pieces to reduce the cost of change [16]. Instead of designing and evaluating the entire system at once, only a small portion of the work is completed at a time to minimize the expense associated with a change in the software requirements. XP improves a software project in five essential ways; communication, simplicity, feedback, courage, and respect. XP consists of some simple rules. The rules are the following :

1. **Planning:** This is the initial phase of the Extreme Programming life cycle, which involves user stories and iterations. The actions that are performed during this stage are:

    a. Writing user stories or user requirements.
    b. User stories are being turned into iterations, each of which addresses a specific aspect of the desired feature's functionality.

c. The programming team is in charge of scheduling each iteration, including time and cost.
d. Each developer registers for iteration

2. **Managing**: Activities of this phase include:

    a. Designate a common area as the team's workspace.
    b. Manageable work speed is established.
    c. Daily stand-up meetings are held.
    d. The velocity of the project is measured.

3. **Design**: The design process starts once the plan has undergone all necessary iterations. The following principles are included in this phase:

    a. Simplicity is achieved by describing something once and without adding functionality.
    b. Select a naming convention or system metaphor that must be consistent.
    c. Using collaboration cards during the design session
    d. Producing spike solutions on basic programs that look for various solutions for a particular problem.

4. **Coding:** The most crucial phase in extreme programming is coding. It takes precedence over all other activities, such as documentation. Principles in this phase include:

    a. Customers must always be available.
    b. Code must be created based on an agreed metaphor or set of standards.
    c. Pair programming is used to create high-quality, cost-effective production code.
    d. Code integration occurs frequently.
    e. A dedicated computer is set up for integration.

5. **Testing:** Testing is integrated into Extreme Programming during the coding and development phases rather than after completion. The principles for this phase are:

    a. To eliminate bugs, codes must undergo unit testing.
    b. All codes must pass unit tests before release.
    c. Customer acceptability testing is done following the customer's requirements, and the developer then provides a report.

6. **Listening:** Extreme Programming allows continual feedback. Listening to customers requires understanding what the customer expects the system to do. It assists developers by making sure they are aware of these needs.

    Figure 1 shows a diagrammatic representation of the Extreme Programming development methodology.

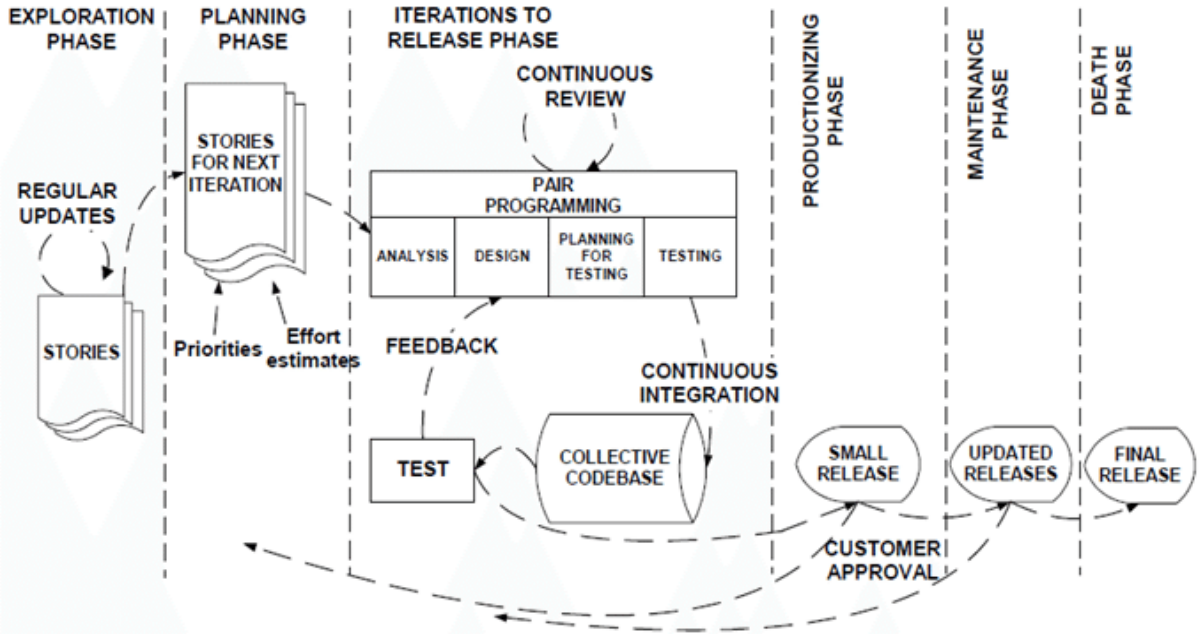

Figure 1: Extreme Programming Development Methodology Life Cycle [9]

XP has several disadvantages, such as a lack of proper documentation, a relatively large time investment as frequent meetings are required, no upfront design document, and a developer's unwillingness to pair programming. However, the merits of this methodology outweigh its limitations. The merits of XP include high quality, continuous testing results in a stable with a small bug rate, low cost, the rapid change allowed at any stage of the software development, continuous feedback from the project owner, and seamless code integration.

2.3. The Shift Left Principle

Shift Left aims to shift activities to the left, that is, to take a more proactive approach to tasks that would otherwise be left until the end and start or move them to the early stages. Shift Left indicates the necessity to change from a reactive to a proactive approach, which reduces time, expenses, and exposure to risks. Many DevOps practices can use Shift Left, an example of which is Shift Left Testing, an approach to moving software testing to the left in the delivery pipeline [3]. Together, these indicate the importance of software development life cycles to accommodate accessibility implementation and take into account various human needs.

2.4. Review of Related Works

2.4.1. Accessibility Inclusion Software Development Process Model

Several accessibility models have been developed, [17]proposed an Accessibility Inclusion Software Development Process model that aims to include accessibility throughout the software life cycle. It is incorporated as one of the development process activities, as well as the ability to support accessibility throughout the software life cycle. The model is based on a hybrid methodology called UCASD—User-Centred Agile Software Development—and contains six essential steps that must be completed sequentially. These phases include analysis, design, coding, deployment, and maintenance. Each phase includes activities

centered on accessibility-related requirements. Figure 2 illustrates the software development process model. This model does not assess the degree to which the developed software conforms to accessibility standards and guidelines or involves an actual test, such as a software development project with actual developers and real requirements.

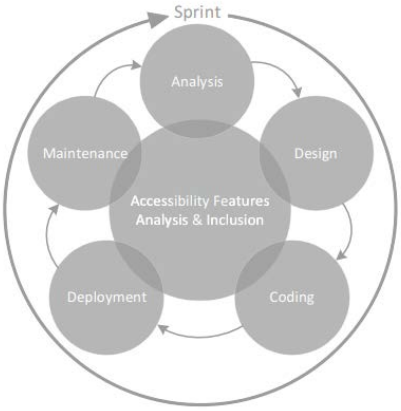

Figure 2: Accessibility Inclusion Software Development Process Model [17]

2.4.2. Website Accessibility Conformance Evaluation Methodology (WCAG-EM)

The World Wide Web Consortium/Web Accessibility in Mind has developed the Website Accessibility Conformance Evaluation Methodology (WCAG-EM) for the evaluation of websites with WCAG 2.0. Website accessibility evaluators can use of this methodology to obtain more reliable results and avoid common errors. Figure 3 shows the stages and activities of the evaluation process that make up this methodology. The stages are not necessarily sequential. In Addition, the exact sequence of the activities carried out during the evaluation stages depends on the goal of the evaluation, the type of website, and the process used by the evaluator. Some of the activities can overlap or be carried out in parallel [18]. WCAG-EM focuses on the evaluation of websites and not accessibility requirements specification, design, or development. It takes a reactive approach and does not support testing while development is ongoing.

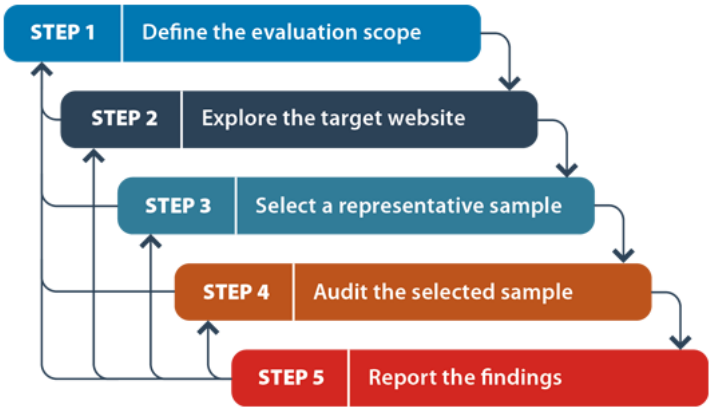

Figure 3: Website Accessibility Conformance Evaluation Methodology (WCAG-EM) [18]

2.4.3. Shift Left a11y Model

It has been demonstrated that accessibility can be incorporated into the development process from the start. The Shift Left a11y Model proposed by [3] aims to find and stop errors at the start of the software development process. It is an iterative technique approach; the phases of this model are sprint, deploy, and next release, with an emphasis on providing accessibility early in the development process, as shown in Figure 4.

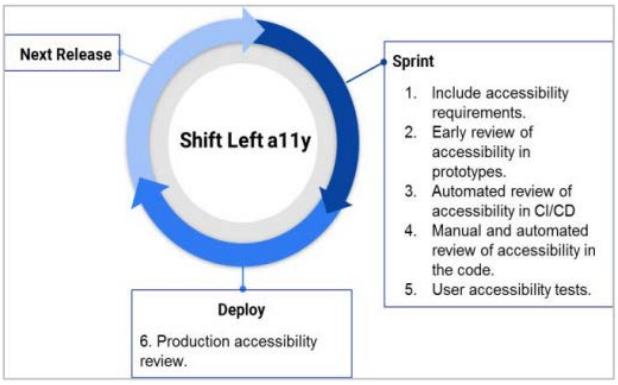

Figure 4: Shift Left a11y Model [3]

1. Sprint: The sprint is the central component of software development, during which a finished, ready-to-use, and possibly release-ready product increment is developed [19]. Right after the preceding sprint, a new sprint starts. The tasks that comprise this phase are accessibility requirements, early review of accessibility in prototypes, automated review of accessibility in Continuous Integration (CI) or Continuous Deployment (CD), automated and manual evaluation of accessibility in the code, and user accessibility tests. These tasks are discussed below.

a. **Inclusion of accessibility requirements**: One of the most important phases in software development is gathering requirements. To move to accessible models, they must meet the following requirements:

   i. **Accessibility Guide:** Web Content Accessibility Guidelines (WCAG) 2.1 offer guidelines for designing software that considers the requirements of each person, as well as those of governments and other organizations. The guidelines are based on four principles: Perceivable, Operable, Understandable, and Robust. Each principle has guidelines that serve as the base and primary goals during development [20].
   ii. **User stories:** These are short statements of the requirements of a client, and acceptance rules should follow. They use natural language, which makes them highly helpful because they help define the issues faced by individuals with various abilities[21]. For example, “As a screen reader user, useful images must have alternate text for me to understand all the essential information on a website.” “As a user with hearing loss, I need subtitles on videos so that I can comprehend the e-learning modules.”

b. **Early review of the prototypes:** At this point, a preliminary review is conducted, which includes looking over upcoming error messages, such as focus state, to help

users understand and navigate where they are. Alternative text to relevant images, and to eliminate any barriers or experiences that may be inaccessible.

c. **Automated review in Continuous Integration/Continuous Deployment:** Code is tested and integrated through the distribution and implementation stages, integrating tests will confirm that the code or content meets accessibility success criteria and that it has been approved for the repository or release to production [3].
d. **Manual and automated code review:** The following methods are used in the manual review: Screen readers are used to identify problems with reading order, focus state, and tab order for keyboard users. Filtering techniques are used to check the color contrast and zoom functionality. Automated review is the process of using tools to determine whether websites meet accessibility guidelines.
e. **User testing:** Manual and automated testing alone are not sufficient to determine accessibility. It is important to test with a group of users with different abilities.

2. Deploy: Despite the possible risks of production-stage testing, it is necessary, [3] recommended using a checklist to guide a regression to quickly identify and ensure accessibility errors are not present. A split test, additionally known as an A/B test, can be conducted to show users two variants of the same webpage to determine which of the alternatives they prefer.
3. Next release: The Next release phase marks the completion of a cycle, which ends with the product's launch. After completing all proposed actions, the cycle is restarted with the next release.

2.4.4. Shift Left a11y Model coupling with scrum methodology

Using the Shift Left a11y model, [3] proposed an integration of the Shift Left a11y model with scrum methodology as a guide for integration into the development project shown in Figure 5. It begins with the presentation of the accessibility requirements, which are then moved to the product stack and then to the sprint stack, which is executed within the iteration. During this cycle that the sprint's automated and manual reviews, as well as the revision of prototyping, are completed, leading to more iterations. The product review in the production stage takes advantage of moving the test to the left. The Shift Left a11y model with scrum methodology serves as a guide to the integration of accessibility into projects; however, it is not flexible to change during the sprint and focuses more on productivity and management than programming and testing, such as unit testing, integration testing, or acceptance testing that can help eliminate any barriers or experiences that may be inaccessible.

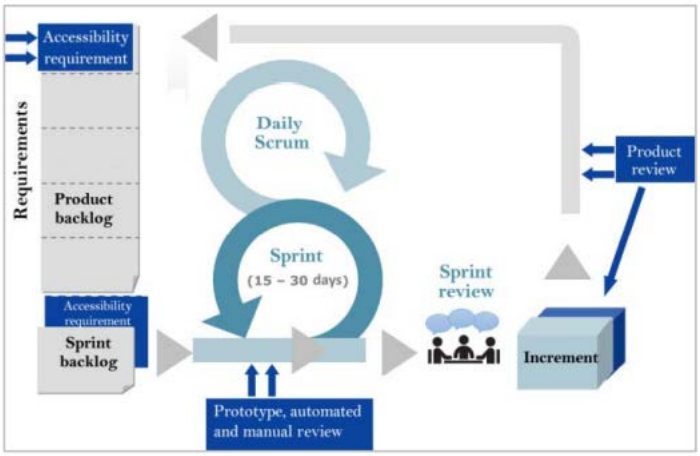

Figure 5: Shift Left a11y Model coupling with scrum methodology [3]

2.4.5. A process model for continuous testing of web accessibility

Much of the research has focused on a reactive approach to handling accessibility issues. [18] developed a process model for continuous web accessibility testing that incorporates three methodologies: the Deming cycle (Plan, Do, Check, Act), WCAG-EM, and Total Quality Management. With each iteration, it delivers feedback and self-feedback. Documents the conclusions drawn from web accessibility issues. Planning the web accessibility evaluation, implementation of the solution, measurment of results, and documentation of the solutions. Since the outcomes of one step serve as the foundation for the next, it is recommended to apply the process model's phases and stages sequentially. Figure 6 presents a process model for the continuous testing of web accessibility. The process model for the continuous testing of web accessibility focuses on the evaluation of websites after development, not at the early stage of development, and does excludes end-users in the evaluation process.

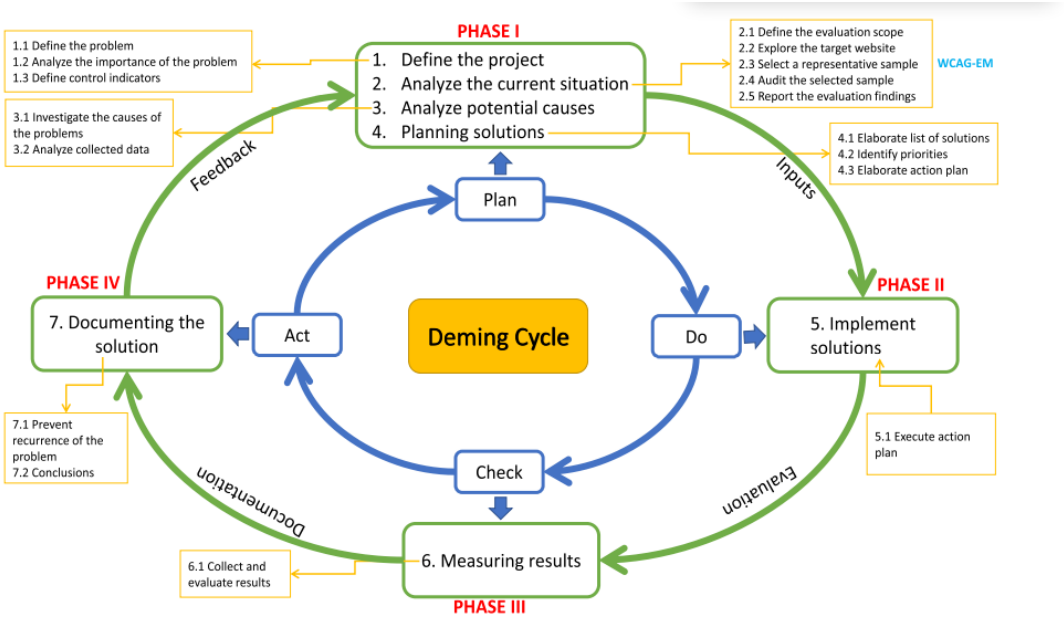

Figure 6: A process model for continuous testing of web accessibility [18]

3. Method

To improve of accessibility in the software development process, the proposed coupling of the Shift Left a11y Model [3] in the Extreme Programming Development Methodology Life Cycle [9]. The coupling is shown in Figure 7. The Shift Left a11y Model with three phases: sprint, deploy, and next release is used in this study to incorporate accessibility from the start of the software development process and to find and stop accessibility errors at their origin. it also accommodates accessibility guidelines and standards. This model tests for accessibility at each stage, beginning on the left side of the development process and ending with an accessible product on the right. which is a gap in the XP development life cycle. This model is coupled with the XP development life cycle, which focuses on programming (this helps to enforce coding standards), and continuous testing, accommodates changes and accentuates customer satisfaction through stakeholder involvement throughout the process. The coupled model makes accessibility a natural part of the process without requiring as much effort as would be required if accessibility guidelines were executed at a later stage of software development such as in the traditional software development process.

4. Proposed Model

In this study, the Shift Left a11y model was coupled with the Extreme Programming development methodology. The Shift Left a11y model is used in this study because it finds and stops accessibility errors at the start of the software development process, which is lacking in the XP development methodology. Extreme programming focuses on programming and continuous testing, accommodates changes, and accentuates customer satisfaction through stakeholder involvement throughout the process. Together, they can make accessibility a natural part of the workflow, reducing the time, effort, and cost of including accessibility in software development. Therefore, this can be used to build more accessible software at a lower cost.

The proposed model, which is the coupling of the Shift Left a11y Model in the Extreme Programming Development Methodology Life Cycle, is ideal for incorporating accessibility early and throughout the software development process while also enhancing testing practices and effective stakeholder collaborations. All phases of the XP development methodology life cycle were adapted, and the Shift Left a11y Model Phases and activities were intertwined to work toward solving the identified issue. The following modifications were made in the exploration phase, which is the initial phase in the XP life cycle, where user requirements, architecture, tools, and technology are defined. Accessibility Guidelines and Standards are added as part of the requirement to develop user stories. After the exploration phase, the next phase is the planning phase. This phase deals with what can be built within the due date, and what is the plan for the next iteration? No modifications were made in this phase.

Iteration to release phases: Tasks such as analysis, design, planning for testing, and testing are implemented by a pair of programmers, one of whom is designated as an accessibility expert. Accessibility checks are included in the continuous review, including automated accessibility review in Continuous integration (CI)/Continuous Deployment (CD) before code is pushed into the collective codebase and automatically deployed to a staging or production environment, manual and automated code review of the software, and user accessibility tests that simulate a real-world interaction scenario are conducted for real user feedback. During the productionizing phase, software deployment occurs in small releases to assess its readiness for production. This includes conducting acceptance tests, system testing, and load testing. To guarantee accessibility in future updates, a production accessibility review was integrated into the productizing phase.

The maintenance phase: Over time, the software undergoes continuous evolution, simultaneously incorporating new functionalities while maintaining the existing ones. This evolution allows for the introduction of new architectural designs, technologies, and future versions of accessibility guidelines. Death phase: This final phase was not modified, serving as the endpoint of the software development process. If the customer expresses satisfaction and has no more stories, it is appropriate to proceed with the system's final release. Alternatively, in cases where the customer demands a set of features that cannot be developed within reasonable economic constraints, it is advisable to close the software development. The coupling of the Shift Left a11y Model in the Extreme Programming Development Methodology Life Cycle is presented in Figure 7

Figure 7: Coupling of the Shift Left a11y Model in the Extreme Programming Development Methodology Life Cycle. Adapted from [9]

5. Discussion

This study presents an improvement in accessibility in the software development process by employing the Shift Left a11y Model, which emphasizes early integration for accessibility, and coupling it with the Extreme Programming Development Life Cycle to effectively incorporate various testing practices and user feedback to enable the development of software that is accessible to all users, including those with impairments.

The sequential steps in a traditional software development process lead to the identification of problems at a later stage because testing is typically conducted toward the end. Other models, such as the WCAG-EM process model for continuous testing of web accessibility[18] take a reactive approach to incorporating accessibility. The Shift Left a11y Model [3] can be used to incorporate accessibility early in the development process; however, it must be incorporated with software development methodology. Another study used the Shift Left a11y model with scrum methodology, which serves as a guide to the integration of accessibility into projects [3]; however, it is not flexible to change during the sprint and focuses more on productivity and management than programming and testing such as unit testing, integration testing, or acceptance testing that can help eliminate any barriers or experiences that may be inaccessible.

The proposed model incorporates accessibility early and throughout the software development process while enhancing testing practices and fostering effective stakeholder collaborations during development.

6. Conclusion and Future Works

The objective set out was to design a shift left-based accessibility model for software development process improvement. This study presents the coupling of the Shift Left a11y

Model and the Extreme Programming Development Methodology Life Cycle. This model serves as a guide for developers in incorporating accessibility into products in the software industry and computer services sectors, making accessibility a natural part of the development process.

In future work, the model will be implemented and the developed software will be evaluated using automated accessibility evaluation tools, and usability testing for several user groups, including those with impairments will be conducted.

References

[1] D. V. Kornienko, S. V. Mishina, and M. O. Melnikov, "The Single Page Application architecture when developing secure Web services," *J Phys Conf Ser*, vol. 2091, no. 1, p. 012065, Nov. 2021, doi: 10.1088/1742-6596/2091/1/012065.

[2] World Health Organization (WHO), "Disability." https://www.who.int/news-room/fact-sheets/detail/disability-and-health (accessed Jun. 08, 2023).

[3] L. Armas, H. Rojas, and R. Renteria, "Proposal for an accessible software development model," *Proceedings - 2020 3rd International Conference of Inclusive Technology and Education, CONTIE 2020*, pp. 104–109, Oct. 2020, doi: 10.1109/CONTIE51334.2020.00028.

[4] Web Accessibility Initiative (WAI), "Introduction to Web Accessibility | Web Accessibility Initiative (WAI) | W3C." https://www.w3.org/WAI/fundamentals/accessibility-intro/ (accessed Jun. 09, 2023).

[5] D. G. Miranda and J. Araujo, "A Framework for Integrating Web Accessibility Requirements in Agile Methodologies," 2020, Accessed: Jun. 08, 2023. [Online]. Available: http://ceur-ws.org

[6] WebAIM, "WebAIM: Web Content Accessibility Guidelines." https://webaim.org/standards/wcag/ (accessed Jun. 08, 2023).

[7] General Services Administration, "IT Accessibility Laws and Policies | Section508.gov." https://www.section508.gov/manage/laws-and-policies/ (accessed Jun. 08, 2023).

[8] A. A. Dror *et al.*, "United by Hope, Divided by Access: Country Mapping of COVID-19 Information Accessibility and Its Consequences on Pandemic Eradication," *Front Med (Lausanne)*, vol. 7, p. 618337, Jan. 2021, doi: 10.3389/FMED.2020.618337/BIBTEX.

[9] F. Anwer, S. Shah Muhammad, U. waheed Waheed, S. Aftab, S. Shah Muhammad Shah, and U. Waheed, "Comparative Analysis of Two Popular Agile Process Models: Extreme Programming and Scrum Agile software development models View project Sentiment Analysis View project," *International Journal of Computer Science and*

Telecommunications, vol. 8, no. 2, 2017, Accessed: Jun. 08, 2023. [Online]. Available: www.ijcst.org

[10] D. M. B. Paiva, A. P. Freire, and R. P. de Mattos Fortes, "Accessibility and Software Engineering Processes: A Systematic Literature Review," *Journal of Systems and Software*, vol. 171, p. 110819, Jan. 2021, doi: 10.1016/J.JSS.2020.110819.

[11] M. Rachmaniah, M. M. Krismanti, and M. I. Darissalam, "Tokocabai marketplace application based on web using extreme programming method," *2020 International Conference on Computer Science and Its Application in Agriculture, ICOSICA 2020*, Sep. 2020, doi: 10.1109/ICOSICA49951.2020.9243214.

[12] WebAIM, "WebAIM: Accessibility Evaluation Tools." https://webaim.org/articles/tools/ (accessed Jun. 09, 2023).

[13] ISO, "ISO/IEC 29138-1:2018 - Information technology — User interface accessibility — Part 1: User accessibility needs." https://www.iso.org/standard/71953.html (accessed Jun. 09, 2023).

[14] ISO, "ISO 9241-171:2008 - Ergonomics of human-system interaction — Part 171: Guidance on software accessibility." https://www.iso.org/standard/39080.html (accessed Jun. 09, 2023).

[15] I. Choudhury, "Agile methods for engineering," *Management for Scientists*, pp. 187–206, Mar. 2019, doi: 10.1108/978-1-78769-203-920191013/FULL/XML.

[16] I. W. W. Pradnyana, A. Fahmi, A. Zaidiah, M. B. Wibisono, B. T. Wahyono, and R. H. Purabaya, "Design and Build an Android-Based Waste Pickup Information System Using the Extreme Programming Method (Case Study: BUMDes Cahaya Buana Paku)," *Proceedings - 3rd International Conference on Informatics, Multimedia, Cyber, and Information System, ICIMCIS 2021*, pp. 142–147, 2021, doi: 10.1109/ICIMCIS53775.2021.9699219.

[17] J. S. e. Silva, R. Gonçalves, F. Branco, A. Pereira, M. Au-Yong-Oliveira, and J. Martins, "Accessible software development: a conceptual model proposal," *Univers Access Inf Soc*, vol. 18, no. 3, pp. 703–716, Aug. 2019, doi: 10.1007/S10209-019-00688-5/METRICS.

[18] M. Campoverde-Molina, S. Lujan-Mora, and L. Valverde, "Process Model for Continuous Testing of Web Accessibility," *IEEE Access*, vol. 9, pp. 139576–139593, 2021, doi: 10.1109/ACCESS.2021.3116100.

[19] A. Hidayati, E. K. Budiardjo, and B. Purwandari, "Hard and soft skills for scrum global software development teams," *ACM International Conference Proceeding Series*, pp. 110–114, Jan. 2020, doi: 10.1145/3378936.3378966.

[20] W3C, "Web Content Accessibility Guidelines (WCAG) 2.1." https://www.w3.org/TR/WCAG21/ (accessed Jun. 08, 2023).

[21] L. Hampson, “How to write user stories for accessibility - TetraLogical.” https://tetralogical.com/blog/2022/05/26/how-to-write-user-stories-for-accessibility/ (accessed Jun. 08, 2023).