Universal Library of Engineering Technology Research Article

ISSN: 3064-996X | Volume 3, Issue 2

Open Access | PP: 36-40

DOI: https://doi.org/10.70315/uloap.ulete.2026.0302007

Mobile User Interface Patterns for Older Adults and People with Disabilities in Accessible Software Engineering

Marchuk Bohdan

Board Member, Boring Owl, Warsaw, Poland.

Abstract

Mobile services increasingly mediate access to essential transactions, yet compliance with accessibility standards alone does not guarantee efficient interaction for older adults and people with disabilities. This article develops a software engineering framework for mobile user interface patterns that links accessibility requirements with measurable interaction cost, implementation choices, and release governance. The framework formalizes focus-step budgeting for assistive-technology navigation and specifies how budgets can be enforced across design systems, accessibility testing, CI/CD gating, and regression monitoring. The analysis synthesizes empirical findings in software engineering, large-scale accessibility evidence, platform guidance, human-centred design standards, and recent work on mobile interaction barriers. A formative evaluation indicates that pattern-aligned interface design reduces navigation burden and improves usability outcomes, supporting focus-step budgeting as an engineering metric rather than a purely descriptive accessibility notion.

Keywords: *Focus-Step Budget, Mobile Accessibility, Software Engineering, Accessibility Testing, Regression Prevention, Screen Readers, Older Adults, Disability, Design Systems, CI/CD.*

INTRODUCTION

Mobile applications increasingly function as transaction-bearing infrastructures that mediate healthcare actions, account recovery, authentication, retail payments, and other flows where interaction friction directly affects completion reliability. For older adults and people with disabilities, breakdowns often arise at the level of traversal efficiency rather than formal access alone. An interface can satisfy accessibility requirements while remaining slow, disorienting, or fatiguing when task completion depends on sequential navigation through an overloaded accessibility tree and a long series of focusable nodes. This discrepancy becomes pronounced in dense, feature-accumulating mobile products, where promotional content, secondary actions, and inconsistent semantics increase the cost of assistive-technology navigation.

This article proposes a software engineering framework that systematizes mobile UI patterns for capability-constrained interaction and introduces focus-step budgeting as a measurable constraint on assistive-technology navigation effort. The intended contribution is practical: convert inclusive interface quality from a pass–fail compliance result into an engineering property that can be defined per task, measured repeatedly, and protected against regression in release pipelines.

MATERIALS AND METHODS

The source base combines empirical software engineering studies of accessibility defects and development practice, platform-level accessibility guidance for iOS and Android, human-centred design standards, and broader accessibility evidence. A. Alshayban, I. Ahmed, and S. Malek [1] document recurring accessibility problems in Android apps and describe persistent failure modes in implementation. Apple Inc. [2] Details iOS accessibility behavior for controls, semantics, and assistive interaction. M. Di Gregorio, D. Di Nucci, F. Palomba, and G. Vitiello [3] analyze how accessible Android applications are built in practice and identify gaps between recommendations and delivery reality. Google LLC [4] provides Android accessibility guidance focused on semantic exposure and focus behavior, while Google LLC [5] specifies Material Design accessibility constraints relevant to target sizing, perceivability, and interaction structure. ISO 9241-210 [6] offers human-centred design principles that support the translation of requirements into engineering processes. WebAIM [7] summarizes large-scale accessibility audits that contextualize persistent barriers. WCAG 2.2 [8]

Citation: Marchuk Bohdan, "Mobile User Interface Patterns for Older Adults and People with Disabilities in Accessible Software Engineering", Universal Library of Engineering Technology, 2026; 3(2): 36-40. DOI: https://doi.org/10.70315/uloap.ulete.2026.0302007.

defines baseline accessibility requirements. S. Yan and P. G. Ramachandran [9] review the state of accessibility in mobile apps and highlight unresolved interaction obstacles. X. Zhang et al. [10] examine the creation of accessibility metadata for mobile applications, reinforcing the centrality of semantics and structure in assistive navigation.

Comparative analysis aligned standards, platform guidance, and empirical findings. Source analysis extracted recurring mobile UI failure patterns and stable engineering responses. Conceptual structuring connected constraint types, interface patterns, and focus-step budgeting. Analytical synthesis produced an integrated framework. Engineering interpretation mapped the framework to design-system governance, test instrumentation, CI/CD gating, and regression control.

RESULTS

Empirical work consistently indicates that compliance does not resolve interaction costs in task execution. Accessibility defects remain common in production software, while traversal order, redundant focus targets, and unstable custom components degrade usability even when baseline requirements are nominally met [1, 3, 9]. Platform guidance and WCAG define semantic exposure, operability, and perceivability requirements [2, 4, 5, 8], yet they do not directly bound the amount of navigation work imposed by an interface during real completion sequences.

A capability-oriented synthesis separates constraints into interacting zones that shape the mechanics of traversal. Visual impairments increase dependence on focus order, semantic grouping, and predictable structural cues [2, 4, 8, 10]. Motor constraints amplify the cost of precision taps, gesture-only controls, and small targets, pushing design toward enlarged targets and explicit actions [4–6]. Cognitive constraints elevate the burden of deep hierarchy and unstable placement, strengthening the value of shallow structure and progressive disclosure [5, 6, 9]. Hearing-related constraints reveal the fragility of audio-only confirmations, while speech-related constraints require equivalent non-voice paths for completion [2, 5, 8].

Focus-Step Budgeting as a Measurable Engineering Constraint

Within these constraint zones, the central engineering problem concerns navigation topology: how many focusable elements a task forces a user to traverse, in what order, and with what recovery cost. Focus-step budgeting operationalizes this by defining a focus step as a single assistive-technology navigation action needed to move to the next focusable element, and then setting a maximum allowable number of steps per task type. The budgeting model adds an engineering layer to standards: WCAG and platform rules specify exposure and operability requirements, while the budget specifies the permitted navigation effort that the task may demand in practice [2, 4, 8].

Pattern Structure and Pattern Clusters

The framework organizes design interventions into four interacting pattern clusters. The first cluster targets navigation economy through shallow hierarchy, stable landmarks, and reduced traversal loops. The second cluster targets input control through enlarged targets, gesture alternatives, and explicit primary actions to reduce accidental activation. The third cluster improves output clarity through semantics-first structuring and redundant feedback, ensuring state changes remain perceivable across modalities. The fourth cluster governs recovery through preventive defaults, direct correction paths, and progressive disclosure to keep error repair within a bounded effort envelope [1, 3, 5, 9, 10].

Threshold Logic and Task Risk

A budget requires task-sensitive thresholds because transactional flows differ in risk, mandatory disclosure, and allowable complexity. High-risk tasks such as payment confirmation, medication confirmation, and critical authentication warrant tighter budgets, as disorientation and delay translate into safety and reliability costs rather than mere inconvenience. Lower-risk flows admit broader limits while still benefiting from controlled structure. This threshold logic links interface design to release criteria by creating measurable acceptance conditions for traversal effort.

Figure 1 illustrates the operational meaning of focus-step budgeting using an annotated traversal example.

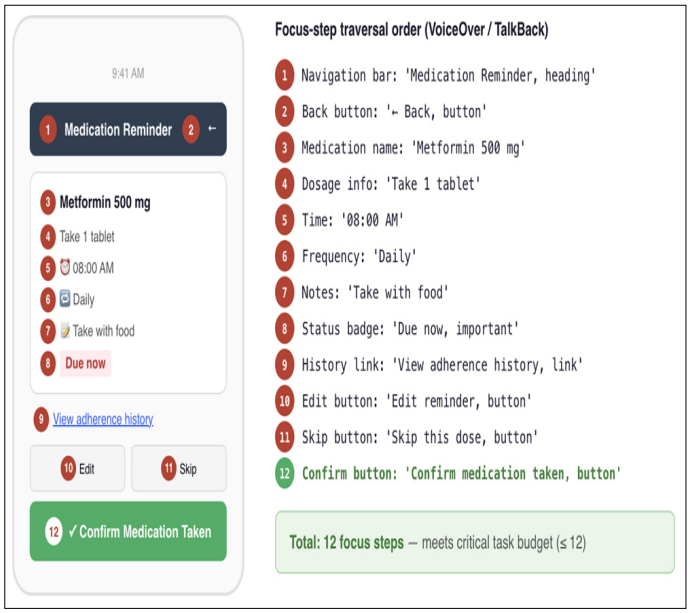

Figure 1. Annotated focus-step counting on a medication reminder screen [2; 4; 8]

The example clarifies why efficiency cannot be inferred from compliance alone: grouping medication details into a single focusable summary and constraining primary versus secondary actions directly reduces traversal length without removing required information. The accessibility tree becomes a countable structure, allowing comparison

between alternative layouts and rapid detection of regressions when new labels or ungrouped elements silently expand the sequence.

Empirical Validation Embedded into the Framework Narrative

A formative evaluation supports the framework by showing that pattern-aligned design reduces navigation cost and improves usability outcomes. Assistive-technology users exhibited a reduction in focus steps from 37.2 to 20.2 (46.2% reduction). For the assistive-technology subgroup ($n = 5$), Wilcoxon signed-rank testing reported $W = 15$, one-sided exact $p = 0.031$, and $r = 0.94$. In the broader sample, task completion increased, completion time decreased, error counts declined, and SUS scores increased. Constraint-specific observations align with the proposed mechanism: ordered announcements reduced backtracking for screen-reader users, enlarged targets reduced accidental activations for motor impairments, shallow hierarchy reduced disorientation for cognitive constraints, and redundant visual confirmation strengthened perception for hearing-related constraints.

DISCUSSION

The framework provides a bridging layer between normative compliance and observed task usability. Empirical studies show that defects and inconsistent implementation practices persist in production apps [1, 3, 9], while platform guidance describes expected component behavior [2, 4, 5] and large-scale accessibility evidence indicates systemic persistence of barriers [7]. The unresolved engineering question concerns bounded navigation effort: how much assistive-technology traversal work a task imposes before completion reliability degrades. Focus-step budgeting answers this by translating semantic structure, hierarchy depth, and recovery design into a measurable constraint that can be enforced in delivery processes.

Figure 2 frames the framework as a mediated mechanism model rather than a checklist: patterns improve outcomes by reducing navigation complexity, lowering motor precision demands, stabilizing perception, and strengthening recovery confidence. This explains cross-constraint benefits, in which one structural intervention improves performance across multiple user groups.

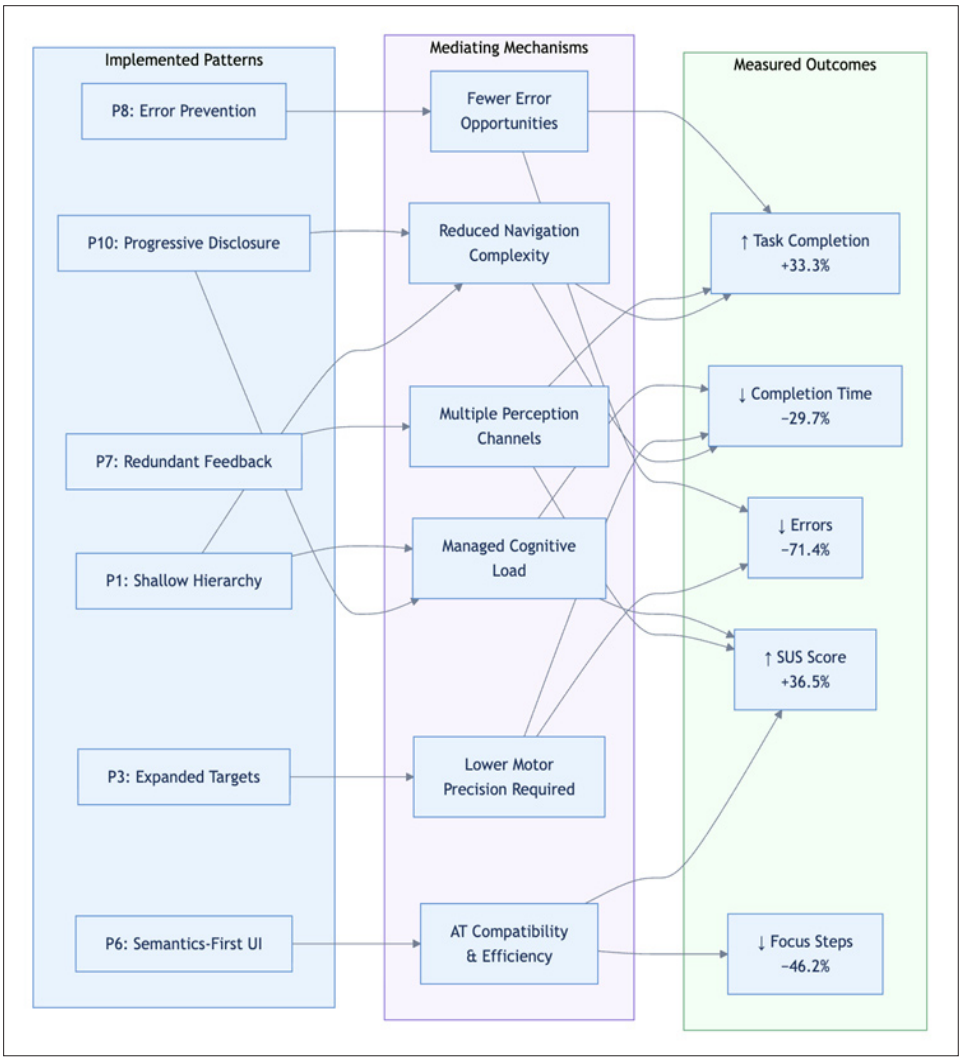

Figure 2. Pattern-to-outcome mapping for implemented interface patterns, mediating mechanisms, and measured outcomes [1; 3; 10]

Engineering adoption requires operational mechanisms compatible with component libraries, test automation, and release governance. Figure 3 presents focus-step budgeting as a release-quality control variable: budgets can be defined for critical tasks, measured automatically from accessibility trees and traversal logs, and enforced through CI/CD gates with regression alerts. This integrates inclusive design into routine engineering accountability rather than isolating it as a late-stage audit event.

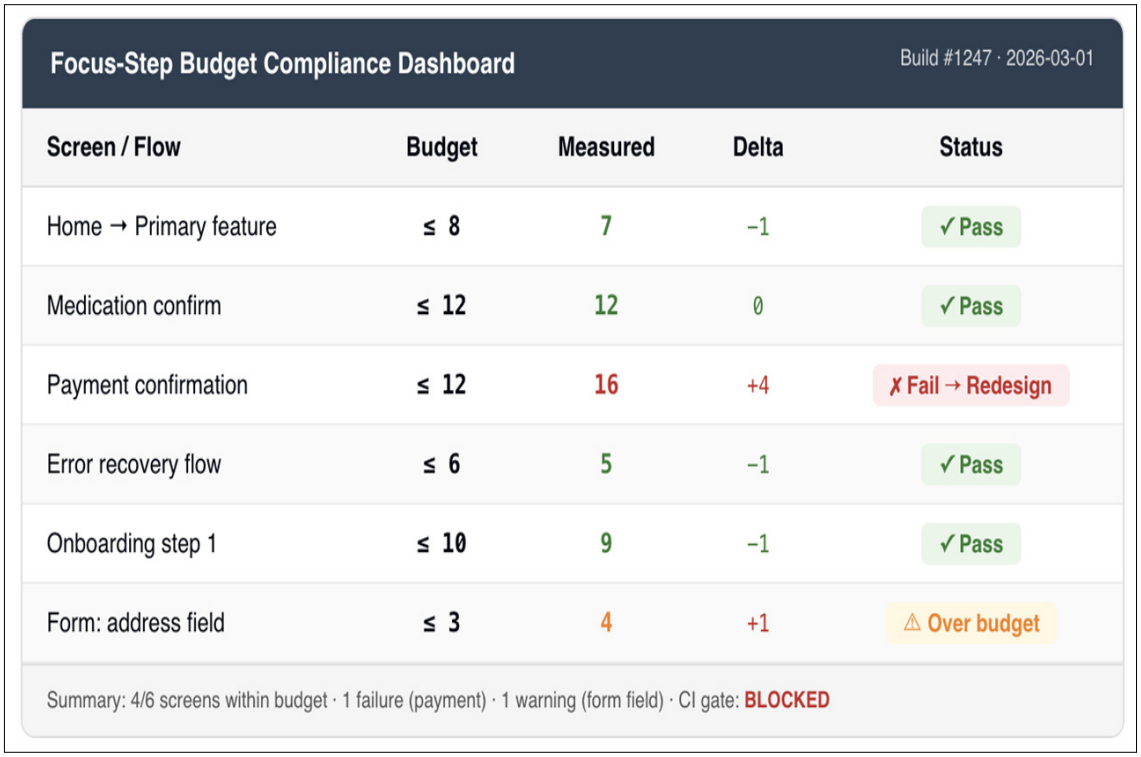

Figure 3. Conceptual compliance dashboard for focus-step budget monitoring in CI/CD [2; 4–6; 8]

At the same time, automated counting does not eliminate the need for assistive-technology testing. Dynamic states, custom widgets, and branch-dependent announcements can produce mismatches between structural metrics and lived interaction, supporting a hybrid model that combines machine-measured traversal cost with targeted manual verification.

The evaluation is designed as a formative study establishing per-task threshold plausibility rather than population-level norms. Task-sensitive thresholds require validation against specific interaction flows rather than generalized sample sizes, which is consistent with task-analytic methodology in HCI research. This approach ensures engineering utility across diverse domains: payment, medication, and authentication flows differ in disclosure constraints and risk exposure, making task-sensitive budgeting more appropriate than fixed global limits.

CONCLUSION

Accessible mobile interaction for older adults and people with disabilities requires evaluation criteria that address the navigation cost of task completion in addition to formal compliance. Focus-step budgeting frames assistive-technology traversal effort as a bounded engineering property that can be defined per task, measured repeatedly, and protected against regression. Pattern-aligned interface design reduces traversal burden and improves usability outcomes, supporting the use of budgets as a practical control mechanism within design systems, accessibility testing, and CI/CD monitoring.

REFERENCES

1. Alshayban, A., Ahmed, I., & Malek, S. (2020). Accessibility issues in Android apps: State of affairs, sentiments, and ways forward. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (ICSE '20)* (pp. 1323–1334). Association for Computing Machinery. https://doi.org/10.1145/3377811.3380392
2. Apple Inc. (2025). *Human interface guidelines: Accessibility*. https://developer.apple.com/design/human-interface-guidelines/accessibility
3. Di Gregorio, M., Di Nucci, D., Palomba, F., et al. (2022). The making of accessible Android applications: An empirical study on the state of the practice. *Empirical Software Engineering, 27*, Article 145. https://doi.org/10.1007/s10664-022-10182-x
4. Google LLC. (2025a). *Android accessibility developer guide*. https://developer.android.com/design/ui/mobile/guides/foundations/accessibility
5. Google LLC. (2025b). *Material Design accessibility*. https://m2.material.io/design/usability/accessibility.html

6. International Organization for Standardization. (2019). *ISO 9241-210: Human-centred design for interactive systems*. https://cdn.standards.iteh.ai/samples/77520/8cac787a9e1549e1a7ffa0171dfa33e0/ISO-9241-210-2019.pdf

7. WebAIM. (2024). *The WebAIM Million and related accessibility surveys*. https://webaim.org/projects/million/2024

8. World Wide Web Consortium (W3C). (2023). *Web Content Accessibility Guidelines (WCAG) 2.2*. https://www.w3.org/TR/WCAG22/

9. Yan, S., & Ramachandran, P. G. (2019). The current status of accessibility in mobile apps. *ACM Transactions on Accessible Computing, 12*(1), Article 3. https://doi.org/10.1145/3300176

10. Zhang, X., de Greef, L., Swearngin, A., White, S., Murray, K., Yu, L., Shan, Q., Nichols, J., Wu, J., Fleizach, C., Everitt, A., & Bigham, J. P. (2021). Screen recognition: Creating accessibility metadata for mobile applications from pixels. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)* (Article 275, pp. 1–15). Association for Computing Machinery. https://doi.org/10.1145/3411764.3445186