

Toward a WCAG-first Responsibility Framework for Treating Accessibility as a First-Class Quality Attribute in Web Systems

Letícia C. M. M. Awawdeh
UFLA
São Sebastião do Paraíso, MG
Brazil
leticia.awawdeh@alumni.ufla.br

Miguel A. F. R. Silva
UFLA
São Sebastião do Paraíso, MG
Brazil
miguel.silva2@estudante.ufla.br

Gabriel A. Moreira
UFLA
São Sebastião do Paraíso, MG
Brazil
gabriel.moreira6@estudante.ufla.br

Sandra S. Rodrigues
UFLA
São Sebastião do Paraíso, MG
Brazil
sandra.rodrigues@ufla.br

Bento R. Siqueira
UFLA
São Sebastião do Paraíso, MG
Brazil
bento.siqueira@ufla.br

Anderson C. Garcia
USP
São Carlos, SP, Brazil
andersongarcia@usp.br

Diego Saqui
UFLA
São Sebastião do Paraíso, MG
Brazil
diego.saqui@ufla.br

ABSTRACT

Web accessibility is often operationalized through interface-level checks, after relevant development decisions have already been made. This work proposes a preliminary responsibility framework that treats accessibility as a first-class quality concern in Web software development, rather than as a post-hoc interface check. The framework maps selected WCAG 2.2 concerns to responsibilities distributed across three modules: semantic data exposure, component contract definition, and architectural governance. Each module is independently adoptable and includes explicit traceability links between accessibility requirements, development decisions, implementation artifacts, and validation evidence. Motivated by the implementation context of a digital Mini-mental State Examination for Children (MMC), the proposal is illustrated through a usage scenario and evaluated through a preliminary traceability-oriented inspection that compares it against a conventional WCAG-as-checklist baseline. The inspection is descriptive rather than causal: it indicates that the proposed framework makes responsibility allocation and accessibility evidence more explicit than the baseline, but it does not involve an implementation, a user study, or a measurement of accessibility outcomes.

KEYWORDS

Digital accessibility, WCAG 2.2, Web software engineering, Responsibility framework, Software architecture, Assistive technologies

1 INTRODUCTION

The development of accessible Web software has been studied with increasing frequency in software engineering literature [5], driven both by normative advances, such as the Web Content Accessibility Guidelines (WCAG) 2.2 [11] and national legislation such as Brazil's Inclusion Law [4], and by the need for digital systems that serve users with disabilities, who represent more than one billion people according to the World Health Organization [10]. Digital exclusion represents an additional barrier to their social, educational, professional, and clinical participation.

WCAG 2.2 organizes Web accessibility around four principles, known by the acronym POUR (Perceivable, Operable, Understandable, Robust), decomposed into testable success criteria [11]. These guidelines are applied primarily at the presentation layer, i.e., to HTML, CSS, and visual components, which is insufficient when accessibility barriers are rooted in earlier development decisions. Translating POUR into development responsibilities means going beyond the interface: designing APIs that expose data semantically, modelling entities that represent users' accessibility needs, defining component contracts that keep compatibility with assistive technologies independent of the presentation layer, and supporting navigation that does not rely exclusively on a mouse or conventional input. Treating accessibility as a first-class quality concern, rather than a post-hoc adjustment, is the assumption motivating this paper – a concern that is particularly acute when the target audience includes children who

rely on assistive technologies from the earliest development decisions.

In this context, this paper proposes a modular, WCAG-first responsibility framework for Web software development, organized into three core modules: semantic data exposure, component contract definition, and architectural governance. Each module is independently adoptable according to the system's requirements and maturity level, and collectively grounded in the POUR principles [11]. The contribution is intentionally narrower than a reference architecture: it organizes responsibilities and traceability links that may support future architectural descriptions for accessibility-critical Web systems.

2 BACKGROUND AND MOTIVATION

The literature identifies accessibility in software engineering as an open research problem. Paiva et al. [5] conducted a systematic literature review covering 94 primary studies published between 2011 and 2019, analyzing how accessibility has been integrated into software engineering processes. Their findings reveal that research efforts have been concentrated primarily on the testing and design phases, while architectural concerns remain less explicit. Notably, the authors explicitly point out that no reference architecture was found in the software accessibility domain, identifying this as a gap and arguing that such an architecture could contribute to solving two recurring problems in the field: the lack of understanding of how people with disabilities interact with technology, and the absence of straightforward methods for developers to implement accessibility features.

This gap is especially relevant to Web systems because accessibility barriers are not produced only by visual interface decisions. They may also emerge from data models, APIs, component contracts, interaction flows, and validation processes that do not preserve evidence of accessibility decisions.

In this context, the MMC is a cognitive assessment instrument designed for children with disabilities, such as speech impairments, motor disabilities and other limitations [2]. Moving from this previously reported digital prototype context to a Web implementation scenario requires accessibility concerns to be addressed before interface-level remediation.

Since the target audience includes children who may rely on assistive technologies to interact with any digital interface, accessibility cannot be treated as a late concern: it must be embedded into the system from its earliest development decisions. Given that the literature offers no reference architecture for accessible software to guide this type of implementation, and that this gap has been explicitly identified as an open research challenge [5], this work proposes a preliminary WCAG-first responsibility framework grounded in the WCAG 2.2 POUR principles, illustrated through an MMC-based Web implementation scenario.

Previous work suggests that accessibility is often treated through guidelines, interface checks, or late validation activities, while the responsibilities that connect accessibility criteria to architectural decisions, implementation contracts,

and verification evidence remain less explicit. This short paper addresses that gap by organizing selected WCAG 2.2 concerns as traceable responsibilities in Web software development.

3 PROPOSED WCAG-FIRST RESPONSIBILITY FRAMEWORK

Study design. The framework was derived conceptually, not empirically: it was built top-down by (i) taking the four WCAG 2.2 POUR principles as the organizing structure, (ii) identifying, for each principle, the development layer at which the corresponding barriers most often originate (data, component, or governance layer), and (iii) aligning the resulting module boundaries with the empirically-grounded, independently-adoptable module strategy proposed by Lu *et al.* [6] for cross-cutting concerns in software architecture. The framework's primary audience is the software architect or technical lead who defines module boundaries and the Accessibility Requirements Specification (ARS) described below; within each module, backend developers are responsible for implementing the data and API contracts (Module 1), frontend/UI developers for the component contracts (Module 2), and the architect or requirements engineer for maintaining the ARS and its traceability links (Module 3). The remainder of this section presents the framework's conceptual grounding, its three modules, and the trade-offs of adopting it.

Reference architectures aggregate reusable architectural knowledge to guide the development, standardization, and evolution of software systems [7, 8]. According to Angelov *et al.* [1], a reference architecture is characterized by a reference model mapped onto software elements and the data flows between them, serving either a standardization or a facilitation goal. In this work, we intentionally describe our proposal as a responsibility framework rather than a complete reference architecture: our contribution is to organize WCAG-derived responsibilities and traceability links that can support future architecture descriptions for accessibility-critical systems. This stance is consistent with the empirically-grounded design strategy advocated by Lu *et al.* [6], in which architectural components addressing cross-cutting concerns, such as accountability and traceability, are modeled as explicit, independently adoptable modules. We apply the same principle to accessibility: rather than treating the POUR principles [11] as presentation-layer constraints, we map selected WCAG 2.2 concerns to responsibilities that can be assigned, implemented, verified, and reviewed. The mapping itself follows Lu *et al.*'s strategy of assigning each cross-cutting concern to a dedicated, independently-adoptable module: for each POUR principle we asked which development layer (data, component, or governance) determines whether the corresponding WCAG success criteria can be met at all, and assigned the resulting responsibility to that layer's module, as detailed next.

From WCAG Criteria to Responsibilities: Figure 1 illustrates the translational path that motivates our proposal. WCAG 2.2 [11] structures accessibility requirements around

four principles: Perceivable, Operable, Understandable, and Robust (POUR), each decomposed into success criteria organized across three conformance levels (A, AA, and AAA). In current practice, however, these criteria are predominantly applied at the presentation layer, remaining disconnected from deeper development responsibilities.

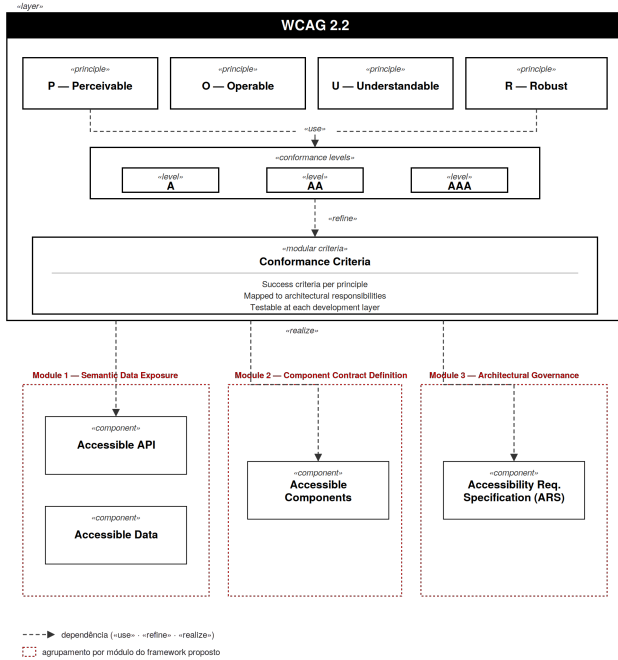


Figure 1: WCAG-first responsibility framework. WCAG-derived concerns are translated into responsibility areas, artifacts, and evidence links across requirement framing, implementation contracts, and verification activities, corresponding respectively to Module 1 (Semantic Data Exposure), Module 2 (Component Contract Definition), and Module 3 (Architectural Governance).

This mapping constitutes the central element of the framework: each POUR principle is translated into a set of modular criteria that must be satisfied not only at the interface level but at relevant development layers, including APIs, components, and data models. In this framework, conformance level is traceable to responsibilities, artifacts, and evidence, instead of being assessed by inspecting rendered HTML alone. Following the model elements identified by Nakagawa *et al.* [7], each module in our framework exposes its own scope, constraints, and architectural decisions, enabling incremental adoption according to system requirements and maturity level.

Framework Structure: The proposed framework is organized into three loosely coupled modules, each addressing a distinct responsibility area and grounded in the POUR principles as illustrated in Figure 1.

Module 1: Semantic Data Exposure. APIs and data models must carry semantically enriched responses, including structured metadata about content type, reading order, status

messages, and interaction semantics. Entities such as *Accessibility Profile* and *Content Descriptor* enable runtime adaptation to diverse user needs, directly operationalizing the Perceivable and Understandable POUR principles at the data layer. This module is analogous to the context engineering component identified by Lu *et al.* [6], which structures the operational context so that downstream components can act responsibly on it.

Module 2: Component Contract Definition. Each interface component must satisfy an explicit accessibility contract specifying required keyboard behavior, focus management, ARIA-related semantics when applicable, error feedback, and assistive-technology behavior, independently of the rendering technology. Decoupling the contract from the implementation preserves the Operable and Robust principles across technology migrations: a concern analogous to the guardrails pattern discussed by Lu *et al.* [6], which enforces behavioral constraints regardless of the underlying rendering layer.

Module 3: Architectural Governance. Accessibility requirements are formalized as an Accessibility Requirements Specification (ARS), serving as the traceability artifact between stakeholder needs and implementation decisions, in line with the architectural decision documentation practices discussed by Valle *et al.* [8]. Automated conformance checks integrated into the CI/CD pipeline operationalize the Robust principle and ensure that each conformance level remains verifiable as the system evolves. However, automated checks should be treated as evidence support rather than as a replacement for human review with assistive technologies and domain stakeholders.

Trade-offs: The framework introduces two main design tensions. **Modularity vs. cohesion:** decoupling the modules reduces technology lock-in but requires explicit synchronization, adding coordination overhead in legacy systems. **Accessibility-first vs. velocity:** front-loading accessibility as a first-order requirement slows initial delivery, but retrofitting compliance post-deployment is substantially costlier [3, 4, 10]. These trade-offs should be interpreted as adoption concerns, not as empirical evidence that the framework produces more accessible systems. As presented, the framework specifies responsibilities and traceability links but not an external artifact catalog, systematic WCAG-to-responsibility translation rules, or supporting tooling; operationalizing it in a development team’s day-to-day workflow.

4 USAGE SCENARIO

To illustrate the proposed framework, consider a Web-based cognitive assessment system derived from the MMC context. [2]. The system exposes a REST API to a Web front-end used by children with different accessibility needs under the supervision of an operator or health professional. Three user profiles illustrate the framework’s application: User 1, who is low-vision and uses a screen reader; User 2, who has a motor impairment and navigates only by keyboard and User 3, who has an attention-related cognitive disability

and is easily overwhelmed by dense interfaces. These profiles were derived from the accessibility needs documented for the assistive-technology users of the source instrument's target population, rather than from a formal requirements-elicitation study conducted for this paper; this is stated explicitly here so the scope of the illustration is not overstated. Whether each user can complete the assessment task depends on architectural decisions made prior to interface implementation.

Module 1 exposes accessibility metadata alongside API data: every response carries a Content Descriptor (content type, reading order, interaction semantics) and an Accessibility Profile resolved once at the gateway and reused across clients. This enables User 1's screen reader to announce assessment content consistently instead of reading disconnected text, operationalizing the Perceivable and Understandable principles [11] at the data layer.

Module 2 attaches a technology-agnostic contract to each UI component, governing keyboard focus and assistive-technology behavior regardless of rendering platform. This enables User 2 to navigate assessment options without focus trapping and to receive a spoken confirmation, the Operable principle, enforced by verification activities before a non-compliant component ever reaches User 2 [11].

Module 3 maintains an Accessibility Requirements Specification (ARS) mapping WCAG 2.2 criteria [11] to architectural decisions and validation items (Table 1), extended before new features ship and re-verified on every pull request. This prevents degradation of User 3's experience as the system evolves, ensuring the Robust principle through planned evidence rather than through ad hoc verification [8], analogous to the continuous risk-assessment patterns of Lu *et al.* [6].

Table 1: Illustrative ARS fragment.

Criterion	Responsibility	Evidence
1.3.1 (A)	Meaningful structure is exposed beyond visual layout	Schema and component review
2.1.1 (A)	Assessment flow is operable by keyboard or equivalent input	Keyboard navigation test
4.1.3 (AA)	Relevant status changes are announced	Assistive-technology review

The modules are independent yet complementary, and may be adopted incrementally. (e.g., Module 3 first, as a governance baseline, with Modules 1 and 2 integrated as the system matures [1, 7]). This scenario is a proof of concept, not a full evaluation; case studies and controlled experiments with development teams are left as future work.

5 PRELIMINARY TRACEABILITY INSPECTION

This feasibility-oriented traceability inspection asks whether the proposed WCAG-first responsibility framework, in which accessibility is treated as a first-class quality attribute distributed across development responsibilities, produces decision material that is more explicit, traceable, and structured with respect to accessibility concerns than a conventional WCAG-as-checklist baseline. It compares two conditions against a single adoption concern: whether organizing accessibility as a first-order development responsibility yields inspectable material that is structurally linked to the POUR principles and explicit about planned validation.

The generic condition (G) represents a conventional architectural baseline in which accessibility guidelines are considered only at the presentation layer, after core architectural decisions are finalized. No explicit mechanism is supplied for tracing accessibility requirements to architectural responsibilities, component contracts, or validation items.

The configured condition (C) represents the proposed responsibility framework, in which accessibility responsibilities are organized across semantic data exposure, component contracts, interaction patterns, and validation mechanisms. Aspects such as implementation quality, technology choices, developer productivity, or the clinical validity of the MMC instrument were not assessed.

Table 2: Conventional WCAG-as-checklist baseline vs. proposed WCAG-first responsibility framework.

Aspect	Generic (G)	Configured (C)
Input	Concern + WCAG criteria	Concern + WCAG criteria + responsibility framework
Responsibility allocation	Concentrated on interface inspection	Distributed across data, components, and governance
Traceability	No explicit link to architectural decisions	Links criteria to responsibilities, artifacts, and evidence
Validation planning	Generic checklist items	Evidence items linked to implementation and review activities

Condition C produces architectural material in which accessibility responsibilities, alternatives, and validation items are linked explicitly to the proposed responsibility framework. Condition G does not include a mechanism linking accessibility concerns to the architectural decisions governing the system. The result is descriptive rather than causal: no implementation was evaluated, no users were observed, no architect independently reviewed the outputs, and no effort or usability measure was collected.

6 THREATS TO VALIDITY

Since the evaluation is descriptive, it inspects structural properties of two architectural conditions rather than outcomes of an implemented system. We organize the threats below following the four categories proposed by Wohlin *et al.* [9]: conclusion, internal, construct, and external validity.

Conclusion validity. The evaluation indicates that Condition C is more explicit and traceable than Condition G, but the finding remains descriptive. It does not show that the proposed framework is correct, that it will produce more accessible systems, or that its traceability mechanisms will hold under real development constraints. No validation item was executed, and no architect independently reviewed the outputs.

Internal validity. The inspection in Table 2 was performed by the authors of the proposed framework, without an independent reviewer, which may reflect the authors' design intent rather than an external assessment. Condition G was characterized from recurring patterns in the literature review, not from direct inspection of a deployed system. Thus, real conventional systems may expose some architectural properties more clearly than assumed, affecting the fairness of the contrast.

Construct validity. The inspection dimensions were defined from reference architecture literature [1, 7, 8] and MMC requirements. Although grounded in established frameworks, they may not cover all aspects of architectural accessibility. The properties in Table 2 are treated as presence or absence of inspectable characteristics, not as measures of depth, completeness, or quality.

External validity. The proposed framework was derived from a single implementation context with demanding accessibility requirements. Its instantiability in other domains, such as public administrative systems, educational platforms, or rehabilitation tools, remains empirically unverified.

7 ETHICAL CONSIDERATIONS

This work does not report a new study with human participants and does not analyze personal data. No new ethics committee approval was required for this architectural analysis. Future empirical studies involving users will follow applicable ethical regulations and LGPD requirements.

8 CONCLUSION

This paper presented a preliminary WCAG-first responsibility framework that organizes WCAG 2.2 POUR principles into architectural responsibilities across semantic data exposure, component contract definition, and architectural governance. Each module defines scope, constraints, architectural decisions, and traceability links among requirements, decisions, components, and validation mechanisms.

The framework was instantiated through the MMC context and illustrated with a Web-based cognitive assessment system involving three user profiles. A feasibility-oriented traceability inspection compared a configured condition, where accessibility responsibilities are distributed across architectural

layers, with a generic condition, where accessibility remains at the presentation layer. The configured condition produced explicit responsibility allocation and traceability to POUR principles; the generic condition lacked a comparable link between accessibility concerns and architectural decisions. The evaluation is descriptive: no implementation was assessed, no users were observed, and no validation item was executed.

Future work includes implementing the framework in the MMC context, refining the ARS with development teams, conducting independent architectural review, and evaluating the resulting Web system with accessibility criteria and users or professionals involved in the assessment context.

GENERATIVE AI USAGE STATEMENT

Generative AI tools were used to support language revision and editorial structuring. The authors reviewed and are responsible for all scientific claims and final content.

REFERENCES

- [1] Samuil A. Angelov, Paul W. P. J. Grefen, and Dirk Greefhorst. 2012. A Framework for Analysis and Design of Software Reference Architectures. *Information and Software Technology* 54, 4 (2012), 417–431. doi:10.1016/j.infsof.2011.11.009
- [2] Letícia Awawdeh, Sandra Rodrigues, Juliana Santos, Rosane Morais, Fernanda Ferreira, Evelyn Corrêa, Ana Pereira, and Amanda Fernandes. 2025. Development of a digital version of the Mini-Mental State Examination for children: a multidisciplinary study. In *Anais do XXIV Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais* (Belo Horizonte/MG). SBC, Porto Alegre, RS, Brasil, 1629–1650. doi:10.5753/ihc.2025.10967
- [3] Len Bass, Paul Clements, and Rick Kazman. 2003. *Software Architecture in Practice* (2 ed.). Addison-Wesley Professional.
- [4] Brasil. 2015. Lei nº 13.146, de 6 de julho de 2015 — Lei Brasileira de Inclusão da Pessoa com Deficiência (Estatuto da Pessoa com Deficiência). Brasília, DF: Presidência da República. Accessed: Jun. 2026. https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm
- [5] Renata Pontin de Mattos Fortes Débora Maria Barroso Paiva, André Pimenta Freire. 2021. Accessibility and Software Engineering Processes: A Systematic Literature Review. 110819 pages. doi:10.1016/j.jss.2020.110819
- [6] Qinghua Lu, Liming Zhu, Xiwei Xu, Zhenchang Xing, Stefan Harrer, and Jon Whittle. 2024. Towards Responsible Generative AI: A Reference Architecture for Designing Foundation Model based Agents. In *2024 IEEE 21st International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 119–126. doi:10.1109/ICSA-C63560.2024.00028
- [7] Elisa Yumi Nakagawa, Flavio Oquendo, and Martin Becker. 2012. RAModel: A Reference Model for Reference Architectures. In *2012 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture*. IEEE, Piscataway, NJ, USA, 297–304. doi:10.1109/WICSA-ECSA.2012.49
- [8] Pedro Henrique Dias Valle, Lina Garcés, Tiago Volpato, Silverio Martínez-Fernández, and Elisa Yumi Nakagawa. 2021. Towards Suitable Description of Reference Architectures. *PeerJ Computer Science* 7 (2021), e392. doi:10.7717/peerj-cs.392
- [9] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, and Anders Wesslén. 2012. *Experimentation in Software Engineering*. Springer, Berlin, Heidelberg. doi:10.1007/978-3-642-29044-2
- [10] World Health Organization. 2023. Disability. Accessed: Jun. 2026. <https://www.who.int/news-room/fact-sheets/detail/disability-and-health>
- [11] World Wide Web Consortium (W3C). 2023. *Web Content Accessibility Guidelines (WCAG) 2.2*. Technical Report. W3C. W3C Recommendation. Accessed: Jun. 2026. <https://www.w3.org/TR/WCAG22/>