PRIM-UI: un método para integrar patrones de interacción en la etapa de requerimientos

Jonathan Delgado Guerrero [1,2][0000-0003-0724-3358]; Leandro Antonelli [3,4][0000-0003-1388-0337] y Diego Firmenich [5][0000-0002-7212-4454]

[1] Facultad de Informática, UNLP, calle 50 esq 120, La Plata, Bs As, Argentina;
[2] Secretaría de Educación Superior, Ciencia, Tecnología e Innovación, Ecuador;
[3] Lifia, Facultad de Informática, UNLP, calle 50 esq 120, La Plata, Bs As, Argentina;
[4] CAETI - Facultad de Tecnología Informática - Universidad Abierta Interamericana;
[5] Departamento de Informática, Facultad de Ingeniería, UNPSJB, Chubut, Argentina
jonathan.delgadog@info.unlp.edu.ar,
lanto@lifia.info.unlp.edu.ar, dafirmenich@ing.unp.edu.ar

Abstract. La fase de requisitos es importante para el éxito de un proyecto de software, ya que un levantamiento bien definido evita costos altos por correcciones y asegura productos alineados con las expectativas del usuario. Sin embargo, el enfoque tradicional suele centrarse en requisitos funcionales y no funcionales, ignorando la interacción del usuario, lo que genera interfaces (UI) poco usables. Este trabajo propone un nuevo enfoque que incorpora patrones de interacción validados en la fase de requisitos para optimizar el diseño de UI. La metodología consta de tres etapas: 1) identificación sistemática de patrones de interacción relevantes para el dominio, 2) análisis contextual de requisitos y 3) mapeo de patrones a requisitos mediante validación heurística. Se evaluó mediante un caso de estudio en una aplicación web para actualización de datos con validación biométrica, con doce requisitos implementados por un equipo de cinco desarrolladores. Los resultados preliminares muestran que los patrones optimizaron la estructura de la UI y la retroalimentación, alineando requisitos con estándares de usabilidad. Los resultados sugieren que el enfoque no solo alinea el diseño con mejores prácticas, sino que también ofrece beneficios medibles en eficiencia y mantenibilidad. Futuros estudios podrían cuantificar su impacto con métricas como SUS y explorar su escalabilidad en sistemas complejos, integrar sistemáticamente las pautas de accesibilidad WCAG, y el uso de aprendizaje automático para recomendación adaptativa de patrones.

Keywords: patrones de interacción, diseño de interfaces, enriquecimiento de interfaces, levantamiento de requisitos, usabilidad.

1 Introducción

La fase de requisitos es importante para el éxito de cualquier proyecto de software por ser el punto de partida del resto de fases. Garantizar que los requisitos estén bien definidos y documentados desde el inicio, podría evitar retroalimentaciones costosas además, reduce el riesgo de entregar un producto que no cumpla con las expectativas del

Proceedings of the 28th Workshop on Requirements Engineering (WER2025),
August 20-22, 2025, Rio de Janeiro-RJ, Brazil.
https://doi.org/10.29327/1588952.28-15

cliente [1]. Contar con un alto nivel de detalle de los requisitos depende del método utilizado para la captación de las reglas de negocio, posteriormente estos son plasmados en documentos o artefactos que serán usados como entradas de las siguientes fases. El proceso tradicional de levantamiento de requisitos en el desarrollo de software ha puesto históricamente un mayor énfasis en la identificación de las funcionalidades que el software debe ofrecer y en las restricciones que debe cumplir. Estos se los conoce como requisitos funcionales y no funcionales. Los primeros responden a la pregunta ¿qué? y se centran en las características que el usuario espera del software. Los segundos responden a la pregunta ¿cómo? y se enfoca en las cualidades del software, como su rendimiento, seguridad y usabilidad [2]. La usabilidad de la interfaz de usuario (UI), definida como la capacidad del sistema para permitir a los usuarios alcanzar objetivos de manera eficiente, es fundamental para resaltar la calidad del software [3].

Aunque estos requisitos funcionales y no funcionales son fundamentales, existe una dimensión importante que a menudo se descuida: la interacción del usuario. Estos se pueden especificar a partir de patrones de interacción, que ofrecen un marco conceptual y un vocabulario compartido. Estos brindan respuestas comprobadas a problemas comunes en el diseño de interfaces de usuario. Cuando los diseñadores utilizan patrones de interacción, pueden hacer que las interfaces sean más fáciles de entender, consistentes, y funcionen mejor. Sin embargo, aunque son útiles, los patrones de interacción no se integran de manera sistemática en los procesos de desarrollo de software [4].

Una de las principales razones de esta brecha es que los patrones de interacción suelen introducirse en fases tardías del ciclo de vida de desarrollo de software (SDLC), cuando gran parte del diseño ya está definido. Esta demora impide que los patrones influyan en las decisiones de diseño desde las etapas iniciales del proyecto. Además, la falta de herramientas y metodologías adecuadas para su integración en la fase de requisitos dificulta su adopción en la industria. Incorporar los patrones de interacción en la fase de requisitos ofrece varias ventajas: asegura la coherencia y calidad de las interfaces desde el principio, reduce el riesgo de errores y omisiones al ofrecer soluciones probadas, y mejora la comunicación entre los actores del proyecto al establecer un vocabulario común [3] [5]. Esto permite a los desarrolladores tomar decisiones de diseño más informadas y alineadas con los objetivos de usabilidad, y facilita la creación de componentes reutilizables, agilizando el desarrollo y reduciendo costos de mantenimiento. En este artículo, presentamos un nuevo enfoque que integra los patrones de interacción en el proceso de levantamiento de requisitos. Nuestra propuesta no solo enriquece la fase inicial del desarrollo, sino que también mejoraría la calidad y consistencia de las interfaces de usuario finales. El artículo se estructura en cinco secciones. La sección 2 presenta los trabajos relacionados, la sección 3 brinda el marco conceptual, la sección 4 describe el método propuesto, y la sección 5 contiene las conclusiones.

2 Trabajos relacionados

La integración de patrones de interacción en la fase de requisitos ha sido explorada mediante enfoques diversos. En la revisión sistemática [6], se han identificado herramientas como WebSpec [7], un lenguaje visual que formaliza requisitos de navegación

e interacción para aplicaciones web, permitiendo la verificación y simulación. La Arquitectura Dirigida por Modelos (MDA) [4], junto con herramientas como UWE, también facilita la separación de *concerns* y optimiza la experiencia del usuario, respaldada por metodologías como el Modelo de Prototipo Intensivo [8]. Es necesario destacar al Modelado de Lenguaje Web (WebMl) [9], que incluye modelos conceptuales como la estructura, composición de páginas y navegación. Como evolución de esta línea de trabajo y promoviendo una mayor estandarización y aplicabilidad, el Grupo de Manejo de Objetos (OMG) propuso el Lenguaje de Modelado de Flujo de Interacción (IFML) [10], con una materialización visible de aplicación en la herramienta *case* llamada WebRatio [11], para dar una especificación precisa del flujo de interacción en interfaces de usuarios anexando conceptos como VistasContenedores, VistaComponentes, Eventos y Flujos de interacción para definir la navegación entre pantallas. En el ámbito de la generación automática de interfaces, Quid [12] se destaca como un lenguaje de dominio específico (DSL) acoplado a un editor WYSIWYG para prototipado web, mientras que el método TRAIN estandariza el diseño de interfaces para aplicaciones de Internet enriquecidas (RIAs). Un DSML propuesto por [13] ha demostrado reducir el esfuerzo de desarrollo entre un 26% y un 77% al trasladar las tareas a la fase de requisitos. Este enfoque se alinea con el modelo sistemático de [14], que integra patrones en fases tempranas mediante la definición de alcance, introspección colectiva y validación iterativa de requisitos. La especificación formal de interfaces se enriquece con artefactos como historias de usuario, maquetas y diagramas WebSpec [15], que clarifican las expectativas de diseño y comportamiento, mejorando la comunicación entre cliente y desarrollador. Las Interfaces Adaptativas (AUI) [16] utilizan aprendizaje automático para analizar interacciones en tiempo real, agrupar usuarios en perfiles y personalizar dinámicamente la interfaz de usuario desde la fase de requisitos. Paralelamente, una biblioteca JavaScript basada en JSON [17] estandariza los requisitos de diseño, reduciendo inconsistencias mediante IA y priorizando patrones de interacción como parte integral del proceso. Además, se ha propuesto una metodología [18] para integrar patrones de usabilidad en la especificación de requisitos, promoviendo la participación activa de los usuarios en el diseño de la interfaz y enfatizando la importancia de considerar la usabilidad desde las etapas iniciales [19].

A pesar de los avances, aún existen desafíos como la integración de patrones en herramientas de modelado y la estandarización de criterios comunes. Además, se requiere mayor evidencia empírica sobre la efectividad de estos métodos en contextos heterogéneos, especialmente en proyectos ágiles o multidisciplinarios. Futuras investigaciones deberán explorar coordinación entre enfoques automatizados (DSML, AUI) y marcos centrados en el usuario (DCU), buscando promover escalabilidad sin afectación de la usabilidad.

3 Antecedentes

La usabilidad de la interfaz de usuario (UI), definida como la capacidad del sistema para permitir a los usuarios alcanzar objetivos de manera eficiente, es fundamental para la calidad del software [3]. Este estudio enfatiza que los requisitos precisos y

enriquecidos, más allá de definir las funcionalidades principales, son fundamentales para alinear el diseño de la interfaz de usuario con las necesidades del usuario. El análisis de requisitos en las primeras fases garantiza la usabilidad al integrar aspectos iniciales sobre la apariencia de la interfaz, las modalidades de interacción y los componentes de manejo de datos. Los requisitos están atados a definiciones funcionales o reglas del negocio que podrían derivar en interfaces de usuario en donde se hace uso de controles o de componentes visuales para el manejo, presentación, captura y procesamiento de datos. Al establecer los requisitos de manera precisa, se puede establecer una relación directa entre las necesidades del negocio, la apariencia, comportamiento de la interfaz y también las potenciales modalidades de interacción entre el usuario y la aplicación a través de los componentes visuales.

En línea con los requisitos para interfaces de usuario, es relevante considerar sistemas de diseños establecidos. Material Design Guidelines (MDG), en su evolución hasta Material 3 (Material You), se ha consolidado como un lenguaje de diseño integral de Google, que ofrece componentes reutilizables, guías de estilo y principios como la personalización y accesibilidad [20]. Además, brinda directrices para la estructura, flujo y elementos interactivos y como estos impactan en la experiencia de usuario [21]. La adopción de tales sistemas de diseño puede ayudar a la selección y adaptación de patrones de interacción sobre las fases iniciales de desarrollo. Paralelamente, las Pautas de Accesibilidad para el Contenido Web (WCAG), desarrolladas por el World Wide Web Consortium (W3C), se han establecido como el estándar internacional para la accesibilidad web [22]. Organizadas bajo los cuatro principios POUR (Perceptible, Operable, Comprensible y Robusto), las WCAG ofrecen criterios de éxito detallados y verificables en diferentes niveles de conformidad (A, AA, AAA).

Los patrones de interacción sobre los componentes, como soluciones probadas y reutilizables, podrían complementar los requisitos al proporcionar guías para el diseño de interacciones comunes. Al relacionar los requisitos con los patrones de interacción, se pueden crear interfaces más consistentes y eficientes, reduciendo el tiempo de desarrollo y mejorando la calidad del producto final.

3.1 Patrones de interacción

Los patrones de interacción optimizan el diseño de interfaces al estandarizar usabilidad y coherencia, siguiendo el uso tecnológico y cerebral [23]. Crean interfaces intuitivas, reduciendo la curva de aprendizaje con elementos reconocibles [24], y se alinean con la Interacción Humano-Computador (HCI) y herramientas como Figma [25, 26]. Minimizan recursos, asegurando consistencia y mantenibilidad. Vinculados al DCU y heurísticas de Nielsen [27], priorizan usabilidad y visibilidad, equilibrando eficiencia técnica y experiencia del usuario [28].

Características. Los patrones de interacción son estructuras que se transforman a lo largo del tiempo, fundamentadas en investigación en usabilidad, HCI y tendencias tecnológicas, esenciales para el diseño de interfaces. Sus características clave incluyen: (1) *Reutilizables*: soluciones genéricas adaptables a contextos múltiples; (2) *Reproducibles*: validados empíricamente; (3) *Centrados en el usuario*: optimizan experiencia y

usabilidad; (4) *Basados en principios*: consistencia, familiaridad y retroalimentación; (5) *Mejora continua*: adaptación a avances tecnológicos y expectativas del usuario. Estas propiedades los consolidan como pilares del diseño de software moderno.

3.2 Clasificación de patrones de interacción

Los patrones de interacción, tal como se describe en los trabajos [23] [26] [29] [30], se pueden categorizar por su nivel de abstracción en: alto (para estructuras generales como la navegación) y bajo (para elementos específicos como los botones). Estos autores también identifican otras categorías relevantes que incluyen formularios, mecanismos de retroalimentación y diversas microinteracciones. La evolución tecnológica ha impulsado además la integración de patrones de índole social (por ejemplo, funciones para compartir o calificar contenido) y aquellos *context-aware* (que utilizan datos del dispositivo). Esta comprensión de los patrones permite vincular los componentes visuales de una interfaz con definiciones de diseño y acción específicas (como un botón dentro de un formulario). En la Tabla 1, se presenta una clasificación y subclasificación de los patrones de interacción, la cual ha sido determinada y adaptada a partir de los conceptos y categorías identificados en los artículos consultados [23] [26] [29] [30] y complementada con catálogos de patrones como el referido en [31].

Tabla 1. Clasificación de patrones de interacción.

Nivel	Clasificación	Subclasificación
Alto nivel	Navegación	Tabs, Estructura/jerarquía, Menú
	Retroalimentación	Wizard, Progreso, Informativos
	Formularios	Validaciones, Contenedores, Eventos, Gestión de datos
	Manejo de datos	Búsqueda, Imágenes
Bajo nivel	Microinteracciones	Social, Wiki, Canales de atención
	Contexto	Ubicación
	Contenido	Paginación, Controles, Presentación, FAQs

3.3 Criterios de selección de los componentes y patrones de interacción

La elección de componentes y patrones de interacción en las fases tempranas del desarrollo, dirigida a ingenieros de requisitos, diseñadores y desarrolladores, se sustenta en criterios formulados a partir del análisis de trabajos como [3], [5] y [16]. Estos criterios incluyen: (1) *Elementos comunes de interacción*, que priorizan la retroalimentación clara, la consistencia visual y la facilidad de aprendizaje para una interacción intuitiva, tal como se destaca en [3] y [5]; (2) *Tecnología y contexto de uso*, que considera la compatibilidad con plataformas y tecnologías específicas (ej. autenticación biométrica, interfaces adaptativas [16]), guiando la documentación de restricciones y la selección de soluciones viables; y (3) *Buenas prácticas y directrices validadas*, que enfatizan la relación con principios de diseño reconocidos, como MDG [20], estándares de accesibilidad WCAG [22] y catálogos de patrones [31], para asegurar soluciones probadas. Estos criterios, derivados de la clasificación de patrones, optimizan la selección de componentes alineados con requisitos funcionales y no funcionales [32].

4 Método propuesto

Este estudio propone un método sistemático para integrar patrones de interacción documentados en la fase de requisitos, abordando la brecha en su incorporación sistemática. El enfoque vincula patrones de interacción con requisitos funcionales y no funcionales, enriqueciendo componentes de UI y garantizando alineación con estándares de usabilidad. Esto podría mejorar la usabilidad, reducir costos de desarrollo (deuda técnica) y acelerar ciclos de entrega.

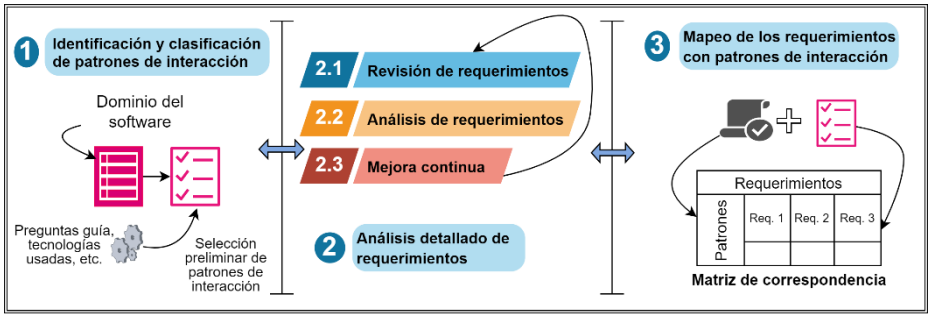

Fig. 1. Método propuesto.

En la Fig. 1, se presentan las etapas propuestas por el método: 1) Identificación y clasificación de los patrones de interacción relevantes; 2) Análisis detallado de los requisitos funcionales y no funcionales; 3) Mapeo de los requisitos con los patrones de interacción.

4.1 Etapas del método

El método propuesto, estructurado en tres etapas secuenciales, busca dar soporte a la implementación y desarrollo de interfaces de usuario mediante: 1) *Identificación de patrones*: Creación de un catálogo de soluciones validadas, filtradas por dominio del software; 2) *Análisis de requisitos*: Comprensión de necesidades del usuario y restricciones técnicas, apoyado en metodologías externas; 3) *Mapeo sistemático*: Relación directa entre requisitos y patrones de interacciones, generando una matriz de correspondencia que asegura coherencia en el diseño y facilita su integración en etapas posteriores del SDLC.

Identificación y clasificación de los patrones de interacción relevantes. Esta fase inicial busca seleccionar patrones de interacción relevantes al dominio del problema, optimizando su integración en fases posteriores. Participan usuarios e ingenieros de requisitos, quienes definen un catálogo estructurado mediante preguntas clave: (1) objetivo del software (tareas críticas del usuario), (2) perfil demográfico y tecnológico de usuarios finales, (3) contexto de uso (entorno, dispositivos) y (4) tecnologías disponibles. Estas respuestas guían la selección de patrones desde un catálogo predefinido [31],

clasificado por nivel de abstracción (alto/bajo), categoría (navegación, formularios) y subclasificación (p. ej., Wizard, pestañas).

Basado en las respuestas, se seleccionan patrones de interacción relevantes de un catálogo derivado de una revisión exhaustiva de literatura. Clasificados por abstracción (alta/baja), tipo (formularios, navegación) y subcategorías (menús, Tabs), la Tabla 2 organiza su búsqueda. Combinando dominio y clasificación, se identifican candidatos para evaluación detallada.

Tabla 2. Ejemplo de preguntas claves de la etapa 1 del método propuesto.

Identificación y clasificación de patrones de interacción – Etapa 1	
Objetivo principal:	Gestión tarjetas (físicas/virtuales) y bloqueo temporal/definitivo.
Características demográficas de los usuarios finales:	Sexo: Todos. Rango de edad: 18 a 50 años Experiencia tecnológica: Alta/Media/Baja
Contexto de uso:	(X) Domicilio (X) Trabajo
Tecnología disponible:	(X) Web ( ) Móviles ( ) Híbrida

Al finalizar esta etapa, se obtendrá un conjunto de patrones de interacción seleccionados que servirán de insumo para la siguiente etapa del método.

Análisis detallado de los requisitos funcionales y no funcionales. Esta etapa sistematiza la revisión de requisitos previamente levantados, enfocándose en aquellos vinculados a patrones de interacción identificados (etapa 1). Su objetivo es garantizar una especificación clara y completa, priorizando requisitos críticos que impactan directamente en la experiencia de usuario (UX) y la coherencia del diseño. El proceso reduce riesgos de errores, detecta inconsistencias (contradicciones, duplicidades) y facilita iteraciones para refinar requisitos, asegurando su claridad y viabilidad técnica. En la Tabla 3 se organizan también conjuntamente con sus objetivos. Sus actividades secuenciales son:

Revisión de los requisitos: Esta actividad revisa la documentación de requisitos para identificar los funcionales (qué hace el software) y no funcionales (cómo funciona), según la metodología previa. Si ya están listados, se toman directamente. Se seleccionan aquellos ligados a la interfaz de usuario, excluyendo, por ejemplo, los de seguridad en bases de datos, priorizando los de mayor impacto en la experiencia del usuario. Por ejemplo, requisitos de seguridad que son de exclusiva aplicación en la base de datos, no serán considerados para las siguientes actividades o etapas. Es fundamental priorizar los requisitos y concentrarse en aquellos que tienen un mayor impacto en la experiencia del usuario.

Análisis de los requisitos: desglosa cada requisito en: (1) Objetivo (meta operativa), (2) Actores (roles beneficiarios), (3) Contexto de uso (escenarios aplicativos), y (4) Patrones asociados (vinculación con patrones predefinidos, etapa 1). Herramientas como diagramas UML de casos de uso y sistemas de trazabilidad de requisitos por ejemplo Jira y Confluence, agilizan este proceso.

Mejora continua: esta actividad incorpora iteraciones para refinar requisitos durante el diseño de UI, adaptando o creando patrones ante ausencia de correspondencia directa.

Este enfoque dinámico garantiza escalabilidad y coherencia con estándares de usabilidad, mitigando desviaciones en fases posteriores del SDLC.

Tabla 3. Actividades y sus objetivos - etapa 2 del modelo propuesto.

Actividades	Tareas	Objetivos
Revisión de requisitos	Revisión de la documentación de requisitos. Identificación de requisitos funcionales y no funcionales. Selección de requisitos relacionados con la interfaz de usuario.	Obtener una visión completa de los requisitos del software. Priorizar los requisitos según su impacto en la experiencia del usuario.
Análisis de requisitos	Análisis detallado de cada requisito. Identificación de objetivos, actores involucrados y contexto de uso. Asociación patrones de interacción/requisito.	Comprender cada requisito. Establecer relación entre requisitos y patrones de interacción.
Mejora continua	Iteración en el análisis de requisitos. Refinamiento de requisitos. Creación/adaptación patrones de interacción.	Asegurar la coherencia y consistencia de los requisitos. Optimizar diseño de la interfaz.

Mapeo de los requisitos con los patrones de interacción. Esta etapa establece una correspondencia explícita entre requisitos funcionales/no funcionales y patrones de interacción predefinidos, asegurando coherencia en el diseño de la interfaz con necesidades de usuario y mejores prácticas. Mediante una matriz de correspondencia, se visualizan relaciones entre requisitos y patrones aplicables, facilitando decisiones en fases posteriores. Este enfoque sistematizado permite desarrollar interfaces sólidas, consistentes y centradas en el usuario, optimizando la eficiencia en el proceso de diseño.

Esta herramienta estructural organiza requisitos (filas) y patrones de interacción (columnas), cuyas intersecciones definen el nivel de conexión mediante una escala Likert 1-4, donde 1 indica *Baja idoneidad*, y 4 representa *Alta idoneidad* para la aplicación del patrón al requisito. En la Tabla 5 se evidencia la matriz de correspondencia resultante del caso de estudio del presente trabajo.

La matriz de correspondencia proporciona una mayor oportunidad de identificar cualquier incoherencia o duplicación en el diseño. Se convierte en un documento de consulta para los diseñadores, quienes en el proceso de trabajo pueden revisar su especificación para tomar decisiones sobre la selección de componentes de la interfaz. Finalmente, la matriz asegura que los patrones (etapa 1) cubra los requisitos suficientes, asegurando así que todos los aspectos funcionales y no funcionales del software, de los requisitos seleccionados en la etapa 2 estén contemplados en el diseño. Es importante destacar que el mapeo de requisitos con patrones de interacción es un proceso iterativo. A medida que avanza el diseño de la interfaz, puede ser necesario revisar y ajustar la matriz de correspondencia. Esto se debe a que nuevos requisitos pueden surgir o los patrones de interacción seleccionados pueden resultar inadecuados. La flexibilidad y la capacidad de adaptación son características que se consideran en esta etapa del método.

5 Evaluación preliminar

Se llevó a cabo un caso de estudio para documentar la aplicación del método propuesto e integrar sistemáticamente los patrones de interacción durante el análisis de requisitos, con el objetivo de brindar soporte para el diseño de interfaces de usuario, mejorar la usabilidad y reducir posibles errores desde etapas tempranas de desarrollo. El contexto del estudio es el desarrollo de una aplicación web destinada a la actualización de datos de afiliados activos (A) y con pérdida de derechos (P) de una empresa del sector financiero ecuatoriano. La aplicación utiliza validación biométrica, diseñada como parte de una campaña de marketing que incluye registro de datos biométricos, promoción de servicios (venta cruzada) y recolección de consentimiento para el manejo de datos privados.

Diseño del caso de estudio. El diseño de este estudio se fundamentó en los principios metodológicos propuestos por [33]. Con base en esta perspectiva, el estudio se diseñó como un caso único (single-case design), justificado por la naturaleza exploratoria y el interés en comprender en profundidad la aplicación del método propuesto por un equipo de $n_d = 5$ desarrolladores ($E_d \in [2,4]$ años de experiencia, $n_{ss} = 3$ semi seniors y $n_s = 2$ seniors), $n_a = 1$ analista funcional para especificar requisitos y $n_g = 1$ diseñador para interfaces de usuario. Se emplearon herramientas de desarrollo (ReactJS v17.0.0, Node.js v18, Azure Pipelines), documentación (Confluence, Figma) y pruebas (Jest v29.4, SonarQube), junto con computadoras portátiles (Lenovo 15AIU7 I7, Dell Latitude 7420), Visual Studio Code v1.92, Git v2.47 y Postman v11.28.3, conformando un entorno adecuado para aplicar el método propuesto. No se incluyó un grupo control, ya que el propósito fue explorar y documentar la ejecución del método en un proyecto específico. La aplicación web cuenta con $Rq = 12$ requisitos ($Rq = 8$ funcionales y $Rq = 4$ no funcionales), detallados en la Tabla 4, que abarcan desde el ingreso de datos hasta la validación biométrica y la gestión de consentimiento.

Tabla 4. Requisitos del software usado para el caso de estudio.

ID	Funcionales	ID	No funcionales
R.1	Ingreso datos básicos afiliado	R.9	Usabilidad: interfaz intuitiva.
R.2	Registro de datos biométricos (reconocimiento facial).	R.10	Accesibilidad: contraste colores, etiquetas ARIA.
R.3	Validación datos con Registro Civil.	R.11	Rendimiento: carga < 2 segundos.
R.4	Promoción servicios (venta cruzada).	R.12	Integridad datos: validaciones
R.5	Reactivación afiliación.		
R.6	Validación celular (SMS/Whatsapp).		
R.7	Validación correo electrónico.		
R.8	Firma consentimiento datos privados.		

Previa a la ejecución del método, se realizó una reunión inicial donde el analista funcional explicó los requisitos al equipo de desarrollo, proporcionando acceso a la documentación en Confluence y a seis servicios externos (srv1-srv6). El diseñador de interfaces de usuario entregó maquetas en Figma como insumos también.

Aplicación del método propuesto. A continuación, se detalla la ejecución de cada etapa que corresponde a la definición del método propuesto.

Etapa 1 - Identificación y Clasificación de Patrones: Para arrancar con esta etapa es necesario conocer particularidades de la aplicación que permitirán más adelante seleccionar los patrones de interacción más adecuados. Se utilizaron las preguntas guías definidas en el método para determinar las siguientes características:

- Dominio del problema: Actualización de datos de los usuarios con validación biométrica (reconocimiento facial).
- Objetivo principal: Captar y validar datos de contacto de usuarios (facial/OTP) para marketing, ofreciendo productos (venta cruzada), reactivando afiliados y obteniendo consentimiento de datos.

Adicionalmente, se levantaron los siguientes datos necesarios como son las características demográficas de usuarios finales: Sexo: Todos; Rango de edad: 23-60 ; Ubicación: Ecuador; Nivel estudio: 3er. Nivel (educación.); Experiencia tecnológica: media.

Basándose en el dominio, objetivo y diseño de la aplicación, se seleccionaron inicialmente patrones de interacción (P.1-formato estructurado, P.2-controles de transformación, P.3-input prompt, P.4-input con retroalimentación, P.5-modales, P.6-notificaciones) del catálogo proporcionado en [31], considerando los criterios de aplicación. Los detalles de cada patrón se encuentran en [34].

Tabla 5. Matriz de correspondencia de requisitos con patrones de interacción.

Requisito	Patrón de interacción	Grado de relación
R.1. Validar cédula del afiliado para iniciar el proceso de validación biométrica.	Formato estructurado	4
	Control de transformación	4
	Input prompt	4
R.2. Utilizar el reconocimiento facial para almacenar fotografía actualizada del afiliado.	Notificaciones	3
R.3. Llamar al servicio de registro civil para validar la fotografía del afiliado capturada.	Notificaciones	3
R.4. Mostrar productos al afiliado para contratación (venta cruzada)	Formato estructurado	4
	Control de transformación	4
	Input prompt	4
R.5. Evaluar la reactivación de afiliados con estado P (pérdida de derechos)	Input retroalimentación	3
R.6. Verificar que el celular ingresado por el afiliado sea válido (OTP).	Input prompt	4
	Input retroalimentación	4
	Notificaciones	4
R.7. Verificar que el celular ingresado por el afiliado sea válido (OTP).	Input prompt	4
	Input retroalimentación	4
	Notificaciones	4
R.8. Recolectar aceptación de manejo de datos privados por parte del afiliado.	Modales	4

Nota*:* el grado de relación en la Tabla 5 se mide en una escala Likert 1-4, donde: 1=*Baja idoneidad del patrón para el requisito*; 2=*Idoneidad moderada*; 3=*Buena idoneidad*; 4=*Alta idoneidad*.

Etapa 2 - Análisis de requisitos: Esta etapa comprende tres actividades principales, las cuales de revisión, análisis y mejora continua de requisitos. El objetivo de la primera es revisar uno a uno los requisitos para determinar si dependen del diseño de interfaz de usuario además de que se establece la prioridad basada en el impacto de este requisito en la experiencia del usuario. En esta actividad los desarrolladores interactúan con el analista funcional y el diseñador para llegar a un consenso en la priorización de los requisitos. En [35] se evidencia el registro de la priorización de los requisitos documentados en la herramienta Confluence. Tras revisar los requisitos, se analizaron aquellos relacionados con la interfaz de usuario (S) para definir su objetivo, actores (usuarios) y contexto (navegador web en PC/móviles), identificando patrones de interacción aplicables de la Etapa 1. Estos patrones, combinables entre sí, se seleccionaron según el diseño de la interfaz y el criterio de aplicación. Por ejemplo, para R.1 (validar cédula), se eligió P.3 (Input Prompt) por incluir un texto de ayuda que guía al usuario sobre el formato del dato. En las columnas Requisito y patrón de interacción de la Tabla 5, se detallan los patrones asociados a cada requisito, destacando su alineación con las necesidades de interacción.

Etapa 3 - Mapeo de requisitos con patrones de interacción*:* en esta etapa se define una matriz de correspondencia, ver Tabla 5, que vincula los requisitos con patrones de interacción y establece su grado de relación.

6 Resultados preliminares

La aplicación del método propuesto en la rama feature-6501-GB arrojó resultados cuantitativos tales como: en la Etapa 1, se identificaron seis patrones (P.1-P.6) adaptados al dominio de actualización de datos biométricos, priorizando formularios y notificaciones. La Etapa 2 revisó los 12 requisitos (8 funcionales, 4 no funcionales), asignando prioridades en Confluence tras tres horas de análisis colaborativo con el equipo. En la Etapa 3, la matriz de correspondencia vinculó patrones como P.3 (Input Prompt) y P.5 (Modales) a R.1 y R.8 con alto grado de relación. El desarrollo totalizó 44 horas (Jira/Tempo), con 39 horas de codificación (3 horas diarias, 13 días), sin vulnerabilidades ni bugs (SonarQube), 8 code smells y 52 minutos de deuda técnica. La cobertura de código alcanzó 48%, con 1662 líneas cubiertas. Estos datos reflejan una implementación coherente alineado a lo que busca proponer el método como contribución.

7 Discusión

La aplicación del método propuesto demostró ser viable en un proyecto real, integrando patrones de interacción que alinearon los requisitos con el diseño de la interfaz desde la fase inicial. Los patrones P1-P3 facilitaron una estructura lógica y accesible, apoyando hallazgos previos sobre diseño centrado en el usuario [1]. P4 y P6 mejoraron la retroalimentación y la gestión de errores, reduciendo ambigüedades en la implementación, como se sugiere en [2]. El uso de modales (P5) en R.8 resultó en una interacción clara para el consentimiento, un aspecto crítico en aplicaciones con datos sensibles. El método ayuda a la estandarización de componentes y alineación temprana entre

requisitos y diseño de la interfaz. Comparando nuestros hallazgos con la literatura existente, la integración temprana de patrones se alinea con la necesidad de enfoque metódicos para derivar diseños de UI evaluables a partir de requisitos validados [37]. También se evidencia que nuestra aplicación práctica guarda relación con la idea de formalización y el uso de patrones de HCI que brindan beneficios para mejorar la comprensión, representación e identificación de interacciones [18]. A diferencia de enfoques de rediseño que se centran en ajustes post-desarrollo, nuestra incorporación temprana de patrones de interacción minimiza correcciones posteriores [38].

8 Conclusiones y trabajos futuros

Este caso de estudio demuestra la aplicación práctica de un método basado en patrones de interacción en la fase de requisitos, destacando su capacidad para guiar el desarrollo de una aplicación web funcional y usable. Los patrones identificados y mapeados, de forma manual, proporcionaron una base estructurada para alinear diseño y funcionalidad, ofreciendo una contribución inicial al refinamiento de procesos de desarrollo de software. El método muestra potencial para integrarse en procesos ágiles, particularmente en proyectos con formularios y validaciones complejas. Sin embargo, se remarcan limitaciones como: el tamaño del equipo, la aplicación de un único dominio específico del estudio y la ejecución manual del mapeo de los patrones con los requisitos.

Como trabajo futuro, se buscará cuantificar su impacto con pruebas de usabilidad (ej. SUS) y análisis de tiempos, además de explorar su aplicación en contextos más amplios. Prioritariamente, se desarrollará una herramienta experimental que implemente la matriz de correspondencia del método PRIM-UI. Esta herramienta permitirá la especificación de los patrones de interacción seleccionados (y alineados con Material Design) utilizando el estándar IFML, sentando las bases para una posterior transformación hacia código y facilitando así la transición del diseño a la implementación. También se integrarán sistemáticamente las pautas de accesibilidad WCAG 2.2+ y se investigará el uso de IA para la selección automática de patrones que cumplan con requisitos funcionales, no funcionales, de MDG y WCAG.

Referencias

1. Amin, T. ul, Shahzad, B.: Improving requirements elicitation in large-scale software projects with reduced customer engagement: a proposed cost-effective model. Requirements Engineering. 29, 403–418 (2024). https://doi.org/10.1007/s00766-024-00425-2.
2. Khan, A.A., Ahmed, F.: Quality Requirements Elicition in Agile. In: 2023 25th International Multitopic Conference (INMIC). pp. 1–6. IEEE (2023). https://doi.org/10.1109/INMIC60434.2023.10466157.
3. Baranauskas, M.C.C., de Almeida Neris, V.P.: Using Patterns to Support the Design of Flexible User Interaction. In: Jacko, J.A. (ed.) Human-Computer Interaction. Interaction Design and Usability. pp. 1033–1042. (2007). https://doi.org/10.1007/978-3-540-73105-4_113.

4. R. A. C. de Souza, R. S. M. de Barros: A Model-Driven Method for the Development of Web Applications User Interaction Layer. In: 2008 2nd IFIP/IEEE. pp. 91–98 (2008). https://doi.org/10.1109/TASE.2008.11.
5. Varma, A.G.: Observing User Interface Design Patterns for Websites from a User-experience Point-of-View. International Journal of Advanced Trends in Computer Science and Engineering. (2020). https://doi.org/10.30534/ijatcse/2020/167922020.
6. Delgado, J., Antonelli, L., Firmenich, D.: Transferencia de diseño de interfaces de usuario a código: revisión sistemática. Presented at (CIIISI'24) , La Habana, Cuba Nov. 25 (2024).
7. Robles Luna, E., Rossi, G., Garrigós, I.: WebSpec: a visual language for specifying interaction and navigation requirements in web applications. Requirements Eng. 16, 297–321 (2011). https://doi.org/10.1007/s00766-011-0124-1.
8. S. Abduljalil, D. -K. Kang: Analysis of human factors in software application design for effective user experience. In: 13th International Conference on Advanced Communication Technology (ICACT2011). pp. 1446–1451 (2011).
9. Ceri, S., Fraternali, P., Bongio, A.: Web Modeling Language (WebML): a modeling language for designing Web sites. 137–157 (2000). https://doi.org/10.1016/S1389-1286(00)00040-2.
10. Brambilla, M., Mauri, A., Umuhoza, E.: IFML: Building the FrontEnd of Web and Mobile Applications with OMG's Interaction Flow Modeling Language. 575 (2014).
11. Acerbis, R., Bongio, A., Brambilla, M., Butti, S.: Model-Driven Development of Cross-Platform Mobile Applications with Web Ratio and IFML. In: 2015 2nd ACM International Conference on Mobile Software Engineering and Systems. pp. 170–171 (2015). https://doi.org/10.1109/MobileSoft.2015.49.
12. Martínez-Ruiz, F.J.: The triad-based design of rich user interfaces for internet applications. In: Proceedings of the 2nd ACM SIGCHI, New York, NY, USA (2010). https://doi.org/10.1145/1822018.1822077.
13. Koch, N., Kozuruba, S.: Requirements Models as First Class Entities in Model-Driven Web Engineering. In: Grossniklaus, M. and Wimmer, M. (eds.) Current Trends in Web Engineering. pp. 158–169. Springer, Berlin (2012). https://doi.org/10.1007/978-3-642-35623-0_16.
14. Bukhari, S.S.A., Humayun, M., Shah, S.M.A., Jhanjhi, N.Z.: Improving Requirement Engineering Process for Web Application Development. 2018 12th (MACS). 1–5 (2018). https://doi.org/10.1109/MACS.2018.8628422.
15. Grigera, J., Rivero, J.M., Robles Luna, E., Giacosa, F., Rossi, G.: From requirements to web applications in an agile model-driven approach. In:, ICWE 2012, Berlin, Germany, July 23-27, 2012. pp. 200–214. Springer (2012). https://doi.org/10.1007/978-3-642-31753-8_15.
16. Rathnayake, N., Meedeniya, D., Perera, I., Welivita, A.: A framework for adaptive user interface generation based on user behavioural patterns. In: 2019 (MERCon). pp. 698–703. IEEE (2019). https://doi.org/10.1109/MERCon.2019.8818825.
17. H. Shah, N. Kamuni: DesignSystemsJS - Building a Design Systems API for aiding standardization and AI integration. In: 2023 (CoNTESA). pp. 83–89 (2023). https://doi.org/10.1109/CoNTESA61248.2023.10384889.
18. Pereira, F.T., Furtado, E.: Integrando Padrões de Usabilidade na Especificação de Requisitos para Apoiar o Re-Projeto Participativo de Interface. In: WER. pp. 131–137 (2006).
19. Raposo, M., Eloy, S., Dias, M.S.: Bridging the gap in customised housing design: Integrating a graphic user interface for user collaboration. PloS one. 19, e0313291 (2024). https://doi.org/10.1371/journal.pone.0313291.

20. Kalac, E., Borovina, N., Boskovic, D.: Preserving interaction design principles while implementing Material Design Guidelines. In: 2021 20th (INFOTEH). pp. 1–6 (2021). https://doi.org/10.1109/INFOTEH51037.2021.9400523.
21. Pinandito, A., Az-zahra, H.M., Fanani, L., Putri, A.V.: Analysis of web content delivery effectiveness and efficiency in responsive web design using material design guidelines. In: 2017 (SIET). pp. 435–441 (2017). https://doi.org/10.1109/SIET.2017.8304178.
22. Caldwell, B., Cooper, M., Reid, L.G., Vanderheiden, G., Chisholm, W., Slatin, J., White, J.: Web content accessibility guidelines (WCAG) 2.0. WWW Consortium (W3C), 5–12 (2008).
23. Tidwell, J.: Designing interfaces: Patterns for effective interaction design. O'Reilly Media, Inc. (2010).
24. Blair-Early, A., Zender, M.: User interface design principles for interaction design. Design Issues. 24, 85–107 (2008).
25. Feng, K.K., Li, T.W., Zhang, A.X.: Understanding collaborative practices and tools of professional UX practitioners in software organizations. In: Proceedings of the 2023 CHI. pp. 1–20 (2023). https://doi.org/10.1145/3544548.3581273.
26. Folmer, E., van Welie, M., Bosch, J.: Bridging patterns: An approach to bridge gaps between SE and HCI. Information and Software Technology. 48, 69–89 (2006).
27. Nielsen, J.: Usability inspection methods. In: Conference companion on Human factors in computing systems. pp. 413–414 (1994).
28. Scott, B., Neil, T.: Designing web interfaces: Principles and patterns for rich interactions. O'Reilly Media, Inc. (2009).
29. Borchers, J.O.: A pattern approach to interaction design. In: Proceedings of the 3rd conference on Designing interactive systems: processes, practices, methods, and techniques. pp. 369–378 (2000).
30. Pan, Y., Stolterman, E.: Pattern language and HCI: expectations and experiences. In: CHI'13 Extended Abstracts on Human Factors in CS. pp. 1989–1998 (2013).
31. Delgado, J., Antonelli, L., Firmenich, D.A.: Descripción de componentes Patrones de interacción. (2025). https://doi.org/10.6084/m9.figshare.29095730.v1.
32. Form Design Patterns, https://ui-patterns.com/patterns/, last accessed 2024/11/04.
33. Yin, R.K.: Case study research: Design and methods. sage (2009).
34. Delgado, J., Antonelli, L., Firmenich, D.A.: Patrones de interacción seleccionados. Etapa 1., https://figshare.com/articles/online_resource/Patrones_de_interacci_n_seleccionados_con_base_en_el_dominio_y_el_objetivo_principal_de_la_aplicaci_n_Etapa_1_/28533194/1, (2025). https://doi.org/10.6084/m9.figshare.28533194.
35. Delgado Guerrero, J.: Priorización de requisitos del caso de estudio, https://figshare.com/articles/dataset/Priorizaci_n_de_requisitos_del_caso_de_estudio/29104691/1, (2025). https://doi.org/10.6084/m9.figshare.29104691.
36. Delgado Guerrero, J.: Especificación del patrón de interacción en el requisito. https://figshare.com/articles/dataset/Especificaci_n_del_patr_n_de_interacci_n_en_el_requisito_descrito_en_Confluence/29104295/1, (2025). https://doi.org/10.6084/m9.figshare.29104295.
37. Silva-Rodríguez, V., Nava-Muñoz, S.E., Martínez-Pérez, F.E., Pérez-González, H.G.: How to select the appropriate pattern of human-computer interaction?. In: (CONISOFT). pp. 66–71. IEEE (2018). https://doi.org/10.1109/CONISOFT.2018.8645944.
38. Walter, M.: An approach to transforming requirements into evaluable UI design for contextual practice-a design science research perspective. In: FedCSIS. pp. 715–724. IEEE (2018).