Integración de Design Thinking, Lean Startup y Scrum para el desarrollo ágil de software: Caso de estudio FESTUM

Integrating Design Thinking, Lean Startup, and Scrum for Agile Software Development: FESTUM Case Study

José Luis Benitez Coronel[1], Ronin David Carrión Carrión[1], Fernanda Soto[1], Daniel Guamán[1]

[1]Universidad Técnica Particular de Loja, Loja, Ecuador.

Cómo citar
J. L. Benitez Coronel, R. D. Carrión Carrión, F. Soto, and D. Guamán, "Integración de Design Thinking, Lean Startup y Scrum para el desarrollo ágil de software: Caso de estudio FESTUM," Ingeniería: ciencia, tecnología e innovación, vol. 13, 2026. https://doi.org/10.26495/2pt9rm12

Información del artículo
Recibido: 06/05/2025
Aceptado: 13/06/2026
Publicado: 02/07/2026

Autor correspondencia
Daniel Guamán,
daguaman@utpl.edu.ec

Este artículo es de acceso abierto distribuido bajo los términos y condiciones de la Licencia Creative Commons Attribution (CC BY)

RESUMEN: Design Thinking, Lean Startup y Scrum son marcos de trabajo ágiles utilizados para la ideación, gestión y construcción de productos y servicios. Los microservicios son considerados como una arquitectura software que permite la descomposición de un sistema en un conjunto de pequeños servicios los cuales son implementados, desplegados y administrados de forma independiente. El objetivo del presente trabajo fue integrar de manera sistemática y práctica los tres marcos ágiles en el proceso de análisis, diseño e implementación de una aplicación móvil denominada FESTUM. Para ello se utilizó la metodología, fases y herramientas provistas por los marcos ágiles. Design Thinking se utilizó para identificar el problema y posible solución siguiendo las etapas de empatía, definición e ideación, apoyados de herramientas como MindMeister, Miro y Jamboard, para el diseño y validación de la interfaz de usuario de los prototipos se utilizó Figma. Los principios de Lean Startup se usaron para construir y evaluar el Producto Mínimo Viable (PMV), para ello se utilizaron herramientas como Mentimeter y Maze, lo que permitió retroalimentación y mejoras iterativas en la aplicación móvil. Para implementar Scrum se utilizó Jira que permite gestionar tareas para construir la aplicación móvil apoyado en tecnologías como Flutter, Dart, Firebase y NestJS. Como resultado se presentó la integración de los tres marcos ágiles que permitieron construir FESTUM, una aplicación móvil con una arquitectura de microservicios que usa un enfoque basado en *Domain-Driven Design* (DDD), desplegada en una infraestructura basada en Docker y Google Cloud con el objetivo de garantizar escalabilidad de la aplicación, y en la que además se integran servicios de Firebase lo que permite funcionalidad, seguridad, interoperabilidad y modularidad. Finalmente, los resultados y pruebas cerradas con usuarios reales evidencian la aplicación sistemática y práctica de marcos ágiles, lo que contribuye a la creación de productos más eficientes, adaptables y orientados a las necesidades reales de los usuarios.

Palabras clave: arquitectura de microservicios, aplicaciones móviles, diseño orientado al dominio., marcos ágiles.

ABSTRACT: Design Thinking, Lean Startup, and Scrum are agile frameworks used for the ideation, management, and construction of products and services. Microservices are considered a software architecture that allows the decomposition of a system into a set of small services, which are implemented, deployed, and managed independently. The objective of this work was to systematically and practically integrate the three agile frameworks into the analysis, design, and implementation process of a mobile application called FESTUM. To achieve this, the methodology, phases, and tools provided by the agile frameworks are used. Design Thinking was used to identify the problem and possible solution following the stages of empathy, definition, and ideation, supported by tools such as MindMeister, Miro, and Jamboard. Figma was used to design and validate the user interface of the prototypes. Lean Startup principles were used to build and evaluate the Minimum Viable Product (MVP), using tools such as Mentimeter and Maze, which allowed for feedback and iterative improvements to the mobile application. To implement Scrum, Jira was used to manage tasks to build the mobile application based on technologies such as Flutter, Dart, Firebase, and NestJS. The result was the integration of the three agile frameworks that allowed the construction of FESTUM, a mobile application with a microservices architecture that uses an approach based on Domain-Driven Design (DDD), deployed on a Docker- and Google Cloud-based infrastructure to guarantee application scalability, and which also integrates Firebase services, allowing functionality, security, interoperability, and modularity. Finally, the results and closed tests with real users demonstrate the systematic and practical application of agile frameworks, which contributes to the creation of more efficient, adaptable products oriented to the real needs of users.

.**Keywords:** microservices architecture, mobile application, domain-driven design, agile frameworks

1. INTRODUCCIÓN

El desarrollo ágil de software debe tener un fuerte compromiso, de modo que los procesos ágiles y sus activos sean conscientes de la sostenibilidad [1], [2]. Sin embargo, el desarrollo de software actual sufre cambios continuos y rápidos debido a la amplia adopción de metodologías ágiles y un rápido time-to-market, lo que puede conducir a aplicaciones poco sostenibles a menos que se adopte un mantenimiento orientado a las dimensiones sostenibles (tecnológicas, económicas, ambientales y sociales) [3]. La integración sistemática de marcos ágiles como Design Thinking [4], Lean Startup [5] y Scrum [6] se fortalecen como una estrategia económica en el desarrollo de soluciones tecnológicas, ya que estos entornos entregan modelos estructurados de ideación, diseño, planificación y construcción del software, facilitando el paso desde la conceptualización hasta la entrega de productos funcionales [7]-[9]. En este trabajo, presentamos FESTUM [10], una aplicación móvil concebida con marcos de trabajo ágiles y diseñada bajo una arquitectura de microservicios y que, al ser desplegada en un entorno en la nube sea funcional y contemple atributos clave de calidad, como el rendimiento, escalabilidad y seguridad. La estrategia integral facilitó un proceso de desarrollo ágil, modular y robusto, garantizando escalabilidad y capacidad de adaptación a distintos escenarios de uso. Además, en el presente trabajo se emplea una metodología sistematizada en un caso práctico centrado en la implementación de aplicaciones móviles escalables a través de la integración de los marcos ágiles que son de gran utilidad en contextos profesionales y que también puede incorporarse como recurso docente en asignaturas vinculadas con el prototipado, ingeniería del software y el desarrollo de aplicaciones empresariales. El resultado de esta investigación también puede ser aplicada en la migración de aplicaciones construidas con arquitecturas monolíticas [11] hacia aplicaciones con entornos basados en microservicios [12], prestando especial atención a aspectos críticos como la gestión de la comunicación entre servicios, los mecanismos de orquestación y las estrategias de despliegue en infraestructuras cloud [13]. Para la investigación, se plantearon tres preguntas que guían el desarrollo del presente trabajo:

- RQ1. ¿Cómo se integran los frameworks para estructurar el desarrollo de software de manera ágil y eficiente?
- RQ2. ¿Cuáles son las fases comunes entre Design Thinking, Lean Startup y Scrum que facilitan la evolución de un MVP hacia una arquitectura funcional basada en microservicios?
- RQ3. ¿Cómo el despliegue mediante contenedores garantiza la escalabilidad y el rendimiento de la aplicación móvil?

El presente trabajo está organizado de la siguiente manera, en la Sección 2 Fundamento teórico, Sección 3 Métodos y materiales, en la Sección 4 Resultados y discusión. Finalmente, en la sección 5 las conclusiones.

2. FUNDAMENTO TEÓRICO

Diversos estudios exponen el impacto de los microservicios en la escalabilidad del software [12], [14], así como la integración de metodologías ágiles en el desarrollo de aplicaciones móviles. Los marcos de trabajo ágil surgieron como una alternativa al enfoque tradicional, permitiendo una mayor flexibilidad y adaptación a cambios durante el desarrollo del software, basándose en ciclos iterativos y entregas incrementales, donde se prioriza la comunicación con el cliente y la mejora continua [15]. Hoy en día, muchas empresas y startups enfrentan retos significativos al escalar sus aplicaciones a la nube, debido a la ausencia de procesos sólidos que combinen buenas prácticas ágiles con tecnologías modernas de arquitectura y despliegue como microservicios y el uso de Docker [16], especialmente cuando se quiere lograr portabilidad y escalabilidad en aplicaciones móviles [12], [17], [18].

2.1 Marcos de trabajo ágil

Design Thinking, propuesto por Tim Brown [7], es una metodología centrada en el usuario que permite la identificación de problemas y la generación de soluciones creativas a través de fases iterativas como empatizar, definir, idear, prototipar y testear. Su uso en el desarrollo de software facilita la alineación de los productos con las necesidades reales de los usuarios [19]. Lean Startup es un enfoque centrado en la experimentación constante de los usuarios sobre una aplicación de software y la validación rápida de hipótesis de una idea de un producto, lo cual se logra a través de un Producto Mínimo Viable (PMV) y la retroalimentación de los usuarios, lo que permite ajustar su desarrollo con agilidad [8]. Scrum ofrece una estructura clara para el desarrollo de soluciones de software a través de iteraciones conocidas como Sprints, lo cual facilita una planificación eficaz y aporta un control preciso sobre el avance del software [9]. Los tres enfoques ágiles expuestos anteriormente, conforman un marco metodológico sólido para abordar el desarrollo ágil de software. Si bien la literatura ha evidenciado que la sinergia entre estos marcos contribuye significativamente para la mejora de la calidad del producto y para la optimización de la experiencia de usuario, la racionalización en el uso de artefactos y herramientas en cada fase del proceso se traduce en una reducción sustancial de los costes operativos y en una notable disminución de los tiempos de desarrollo más aún si se consideran atributos de calidad como la escalabilidad, la modularidad, la mantenibilidad y el rendimiento del software [13]. Aunque se han logrado avances significativos en la adopción de metodologías ágiles dentro del ámbito tecnológico, aún existe una brecha importante cuando se trata de integrar de manera coherente enfoques ágiles dentro de un flujo de trabajo integral. Esta falta de articulación muchas veces no permite una transición fluida desde la ideación inicial hasta la implementación técnica, lo cual es un desafío en contextos que requieren arquitecturas basados en microservicios. En consecuencia, es necesario avanzar hacia una comprensión más profunda de estrategias metodológicas que permitan una integración armónica y eficiente de los marcos de trabajo ágil, desde una perspectiva holística orientada tanto a la generación de valor como a la mejora del rendimiento del proceso de desarrollo [17].

2.2 Microservicios y contenedores

Los microservicios son servicios pequeños y autónomos que trabajan juntos. En aplicaciones monolíticas, el código crece a medida que se agregan nuevas funcionalidades, con el tiempo uno de los problemas puede estar asociado a la calidad y mantenibilidad del software debido a que aumenta su complejidad y deuda técnica. Los microservicios a diferencia de los monolitos son totalmente independientes entre sí [20]. Una de las ventajas de los microservicios es que las diferentes partes de una solución de software pueden estar conformadas por distintas tecnologías independientes, pero que trabajan de manera eficaz [21]. Otro aspecto importante que mencionan los autores respecto de los microservicios es que se mejora la capacidad de respuesta ante fallos, así como su escalabilidad. Por otro lado, un contenedor es la unidad estándar de software que empaqueta el código y todas sus dependencias para que la aplicación se ejecute de forma más rápida y segura entre diferentes ambientes [16]. Un contenedor puede ser considerado como una especie de virtualización de aplicaciones en un ambiente estándar que permite convivir a diferentes contenedores, cada uno con una aplicación construida en diferentes tecnologías, en un mismo equipo anfitrión. Como se menciona en AWS [22], se utiliza para ejecutar aplicaciones basadas tanto en Linux como en Windows. Para ello, los contenedores aíslan de su entorno a la aplicación instalada y se garantiza que su funcionamiento sea correcto. Como se detalla en la documentación oficial de Docker [16], un contenedor es la unidad estándar de software que empaqueta el código y todas sus dependencias para que la aplicación se ejecute de forma más rápida y segura entre diferentes ambientes. Entre las buenas prácticas en el diseño de microservicios constan el bajo acoplamiento y alta cohesión, lo que permitirá que las modificaciones realizadas en

un servicio no afecten a los demás y así mantengan su independencia, eliminando la necesidad de desplegar nuevamente todo el sistema o gran parte de este [21].

3. MÉTODOS Y MATERIALES

La metodología en el presente trabajo usa los principios y artefactos de los 3 marcos de trabajo ágil (Design Thinking, Lean Startup y Scrum) a través de los cuales se diseña e implementa una aplicación móvil denominada FESTUM la cual tiene como objetivo la organización de eventos. Durante las primeras fases de la implementación de los marcos ágiles, se identifica la falta de organización, interacción y personalización como uno de los problemas que enfrentan los organizadores de eventos.

La integración de los marcos ágiles para construir la aplicación móvil de manera ágil y eficiente se visualiza en la Figura 1, donde por cada fase se realiza la trazabilidad para gestionar el proyecto y desarrollar el producto de software.

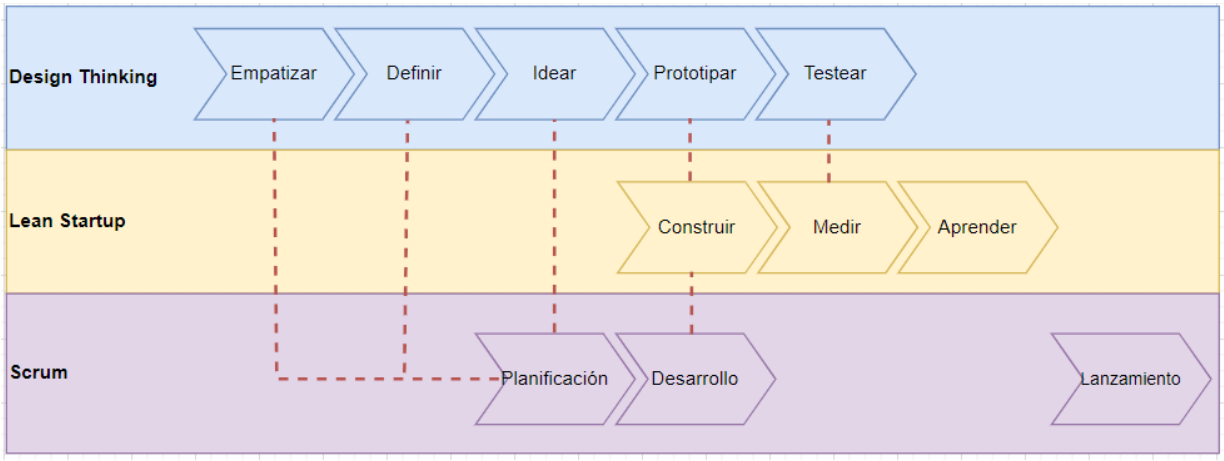

Figura 1. Framework Ágiles y su integración. Fuente: elaboración propia.
Fuente: elaboración propia.

En la Figura se evidencia que las fases de Design Thinking, Lean Startup y Scrum permiten articular un flujo de trabajo ágil y coherente. La etapa de Empatizar en Design Thinking se alinea estrechamente con la fase de planificación en Scrum, ya que ambas buscan comprender el contexto y preparar el terreno para el desarrollo. Del mismo modo, las fases de Definir e Idear se relacionan con esa misma etapa de planificación, en la medida en que se delimitan los objetivos y se empieza a perfilar lo que se va a construir. Posteriormente, la fase de Prototipar se conecta con la construcción en Lean Startup como con el proceso de desarrollo en Scrum, lo que permite dar forma a ideas concretas mediante prototipos o versiones funcionales del producto. La etapa de Testear en Design Thinking se relaciona con las fases de medición y aprendizaje de Lean Startup, ya que su objetivo es evaluar la experiencia del usuario y realizar retroalimentación para futuras mejoras. Finalmente, el lanzamiento en Scrum representa la finalización del ciclo. En el caso de FESTUM, la sinergia entre los marcos de trabajo ágil ha sido clave para diseñar una aplicación móvil empresarial robusta, estructurada y con un potencial comercial real. Las herramientas y artefactos generados en cada una de las fases de los marcos ágiles propuestos, desde la fase de ideación hasta la implementación de la aplicación móvil se exponen en la Figura 2 y se detallan a continuación.

Figura 2. Frameworks y herramientas. Fuente: elaboración propia.

3.1 Fases

3.1.1 Fase Empatizar - Mapa de Propuesta de valor

En Design Thinking la fase de empatizar hace uso del mapa de la propuesta de valor para alinear las ideas iniciales de FESTUM con las necesidades y deseos de los usuarios finales. Este mapa combina el perfil del cliente, que incluye sus alegrías, frustraciones y trabajos. Nuestra propuesta de valor se divide en dos partes principales: el Mapa de la propuesta de valor y el Perfil del cliente (Segmento), estructurados para analizar y alinear las necesidades del cliente con los servicios o productos ofrecidos por una empresa (Ver Figura 3). El Mapa de la Propuesta de Valor se representa de forma visual como un cuadrado dividido en tres secciones: creadores de alegrías, productos y servicios, y aliviadores de frustraciones. Cada una se representa por un ícono distintivo: un regalo, una campana de servicio y una cápsula, respectivamente que facilita la interpretación del modelo. En frente del cuadrado se sitúa el Perfil del Cliente, representado por un círculo segmentado en tres partes: alegrías, trabajos del cliente y frustraciones, lo que permite conectar de manera clara las necesidades y expectativas del usuario con las soluciones ofrecidas.

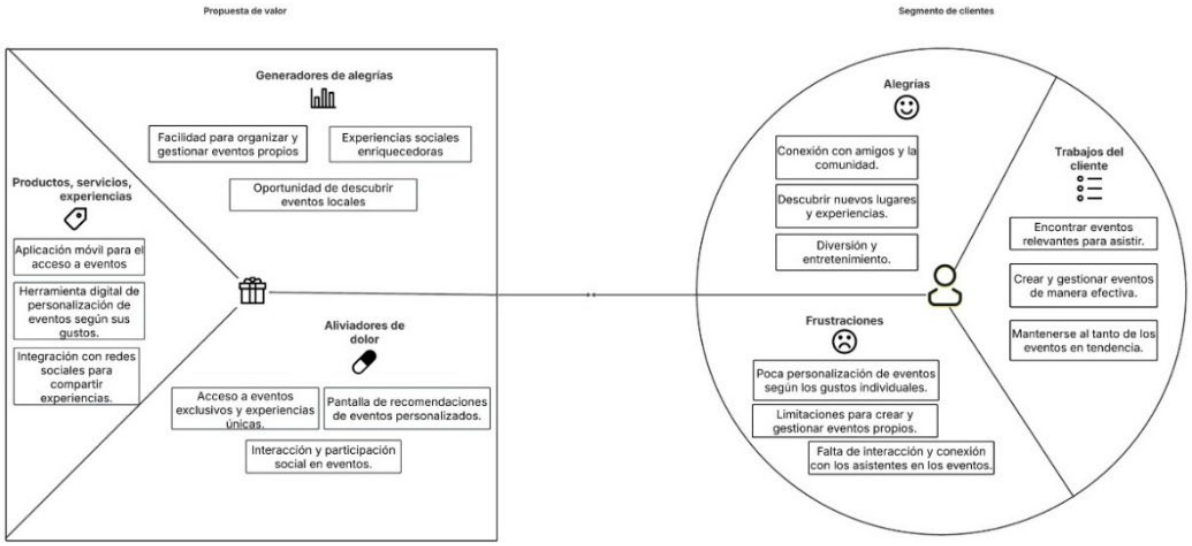

Figura 3. Ejemplo de mapa de propuesta de valor.
Fuente: elaboración propia.

3.1.2 Fase definir – Proto-Journey

El Proto-Journey es una herramienta fundamental para el desarrollo de la aplicación móvil ya que permite definir lo que se conoce como el viaje del usuario a través de sus comportamientos e interacciones. Este proceso incluye una declaración del problema y la priorización de ideas para encontrar soluciones efectivas. En la Figura 4 se presenta un escenario como “el caso de Luis José, un usuario que quiere organizar su fiesta de cumpleaños, pero se encuentra con varios obstáculos debido a su apretada rutina universitaria. A pesar de contar con la ayuda de sus amigos y utilizar redes sociales para coordinar el evento, no logra concretar la celebración tal como la había imaginado”. Esta figura presenta varios puntos de dolor clave en la experiencia del usuario, y al mismo tiempo revela oportunidades para intervenir con soluciones más efectivas. A través del uso de Proto-Journey, se identificó un patrón común en el cual muchos usuarios experimentan dificultades similares al intentar planificar eventos por su cuenta. Este hallazgo fue fundamental para detectar necesidades críticas que la aplicación debe resolver con el objetivo de brindar una experiencia más sencilla, ágil y útil.

3.1.3 Fase idear – Brainstorming

Se realizaron múltiples sesiones de brainstorming colaborativas con la intención de generar ideas creativas y factibles. Para asegurar una exploración amplia y no limitarse a las primeras ideas, se utilizaron herramientas como el Abundance Canvas, que permitió ampliar la mirada sobre las posibles soluciones. En esta fase las ideas fueron agrupadas, discutidas y priorizadas, lo que llevó a identificar una propuesta con gran potencial de impacto y viabilidad, donde su resultado es una solución apropiada para abordar los problemas detectados y generar un valor real y tangible en “una aplicación móvil especializada en la gestión de eventos”. Como parte del proceso de brainstorming, se llevaron a cabo entrevistas con usuarios las cuales fueron diseñadas para abordar temas clave como los tipos de eventos que suelen frecuentar, sus métodos de pago preferidos y los desafíos más comunes que enfrentan al momento de organizar celebraciones (Ver Figura 5). Los insumos generados en estas dinámicas se convierten en la base sobre la cual se seleccionan las ideas más prometedoras, aquellas que pasarán a formar parte de las funcionalidades esenciales que debía incorporar la aplicación móvil.

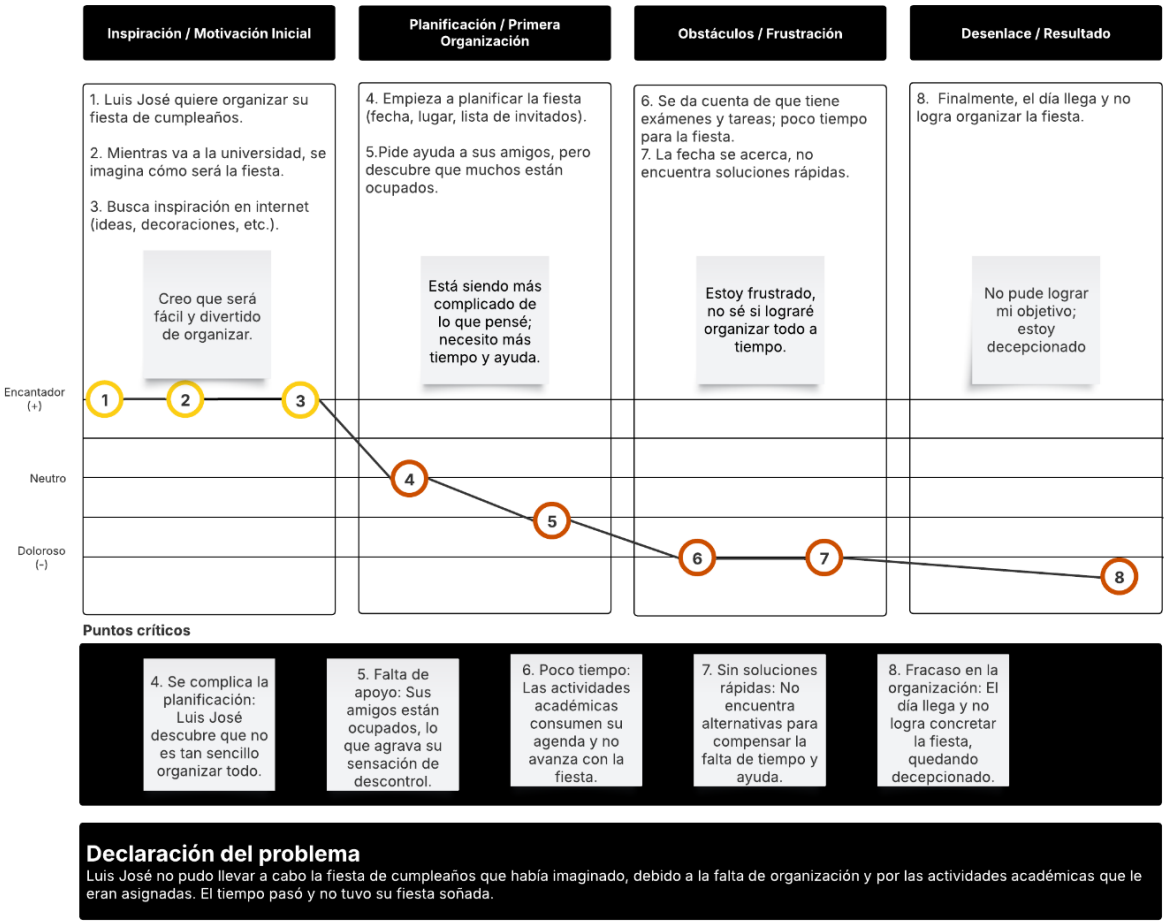

Figura 4. Ejemplo de Proto-Journey. Fuente: elaboración propia.

Entrevistado/ Preguntas	Christian Pugo	Luis Chocho	Ariadna Aleaga	Claudia Encalada
1. Usted en su tiempo libre, hablando de fines de semana, ¿Sale a eventos fiestas reuniones?, ¿Le cuesta decidir acerca de dónde ir?	En mi tiempo libre, especialmente durante los fines de semana, suelo salir a eventos, fiestas y reuniones para mantenerme al tanto de las tendencias y novedades en el mundo de los eventos. A veces me cuesta decidir dónde ir, ya que hay muchas opciones interesantes y quiero asegurarme de elegir aquellas que me aporten conocimientos y contactos relevantes para mi trabajo.	En mi tiempo libre, especialmente los fines de semana, me gusta salir a eventos, fiestas y reuniones con mis amigos. A veces, puede ser difícil decidir dónde ir, ya que hay muchas opciones y a veces no estamos seguros de qué evento será el más divertido.	1. No mucho, pero me cuesta cuando tengo más compromisos	Salgo cuando tengo planes ya hechos, prefiero no hacerlos yo porque ahí si se vuelve muy tedioso porque soy muy indecisa.
2. ¿A qué tipo de eventos, fiestas normalmente asiste?	Normalmente organizo una variedad de eventos, desde fiestas temáticas y cumpleaños hasta bodas elegantes y eventos musicales. También puedo organizar lanzamientos de productos y conferencias, dependiendo de las necesidades de mis clientes.	Normalmente, asisto a fiestas en casa de amigos, conciertos locales, eventos deportivos y ocasionalmente a algún bar o club nocturno. Me encanta la variedad, así que trato de mezclar diferentes tipos de eventos.	Familiares y fiestas entre amigos	Fiestas con amigos o familia.

Figura 5. Entrevista a usuarios. Fuente: elaboración propia.

3.1.4 Fase prototipar – Woreframe /Mockup y prototipos

La Figura 6 presenta los prototipos de baja fidelidad desarrollados para FESTUM. Estos prototipos, que incluyen bocetos y *wireframes*, permiten visualizar de forma preliminar los elementos de interfaz de usuario (UI) de la propuesta, con el objetivo de obtener

retroalimentación temprana por parte de los usuarios. A través de herramientas tecnológicas se diseñaron interfaces simples pero funcionales, enfocadas en facilitar la organización de eventos. Algunas de las funcionalidades diseñadas son la gestión de invitados y la compra de entradas, integradas en un flujo de navegación intuitivo que busca mejorar la experiencia del usuario desde las primeras interacciones.

Figura 6. Prototipo de baja fidelidad usando Balsamiq. Fuente: elaboración propia.

Validación con usuarios: Para validar los prototipos se seleccionó a un grupo de usuarios con el objetivo de obtener retroalimentación directa sobre la funcionalidad y la usabilidad de la aplicación, donde se observó la interacción de los usuarios en las distintas interfaces de usuario, de igual forma se recolectó comentarios y sugerencias que permiten mejorar el diseño de la aplicación FESTUM de acuerdo con las expectativas y necesidades de los usuarios.

Iteraciones Basadas en feedback: A partir del feedback recibido, se llevaron a cabo múltiples iteraciones de los prototipos, cada una orientada a perfeccionar la experiencia del usuario y resolver los problemas detectados durante las pruebas. Estos ajustes no fueron menores: se hicieron cambios en la navegación, se reorganizó la ubicación de funciones clave y se optimizó el flujo general de la aplicación para hacerla más intuitiva y accesible. Cada ciclo de mejora aportó un nuevo nivel de refinamiento, acercando la solución a lo que realmente esperan y necesitan los usuarios (Ver Figura 7).

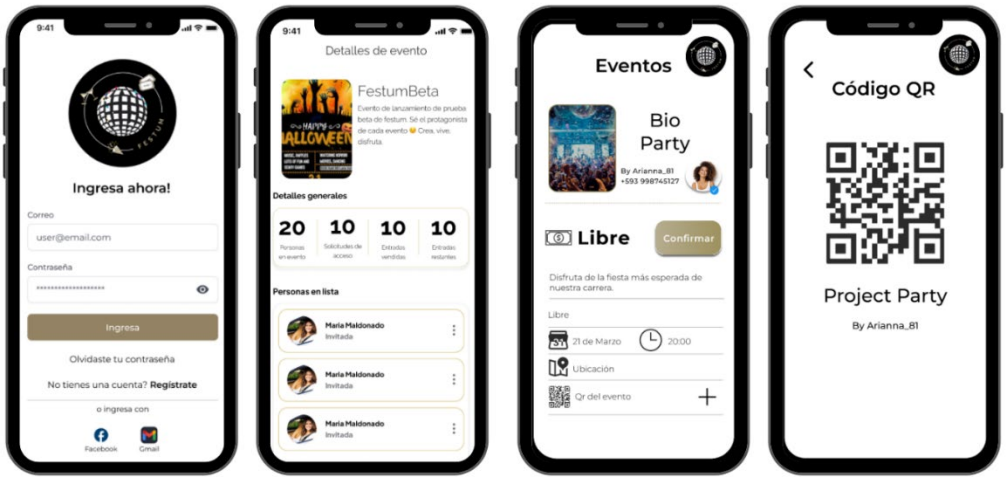

Figura 7. Prototipo de alta fidelidad usando Figma. Fuente: elaboración propia.

Luego de la validación y varias iteraciones en base al feedback de los usuarios se incluyen estos cambios en el desarrollo de un prototipo de alta fidelidad. Esta etapa permite tener un mayor detalle en el diseño, simular el comportamiento e interacción del usuario con la aplicación móvil y optimizar elementos antes y durante el desarrollo (Ver Figura 8).

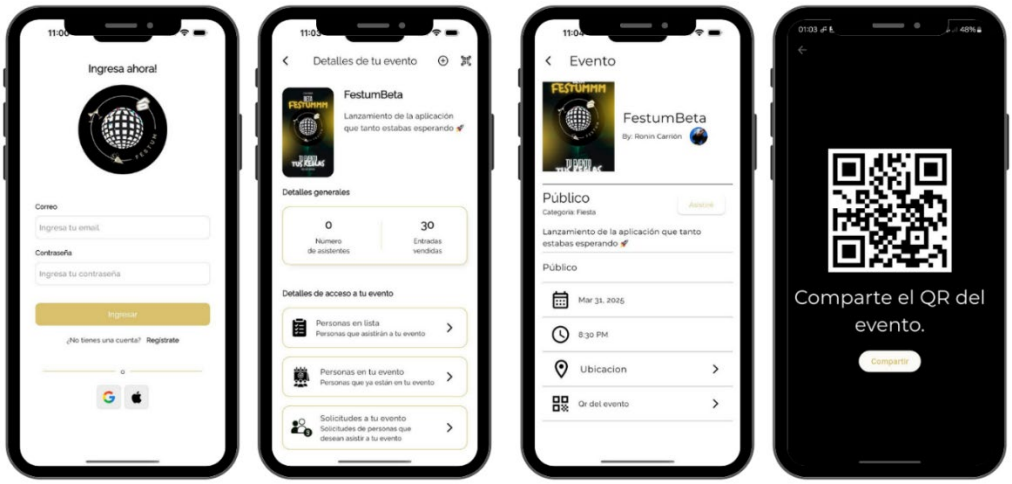

Figura 8. Aplicación FESTUM desplegada en un dispositivo móvil. Fuente: elaboración propia.

Las fases de planificar, construir, medir y lanzamiento presentan el resultado de la integración de los tres marcos ágiles lo que permite implementar y automatizar el pipeline para construir FESTUM como aplicación móvil que usa una arquitectura de microservicios y que está basado en Domain-Driven Design (DDD).

3.1.5 Fase planificar / construir /desarrollo

Para la construcción de la aplicación se utiliza el enfoque Mobile DevOps configurando las herramientas y las conexiones entre ellas, siendo notable la integración con GitHub Actions (Ver Figura 9).

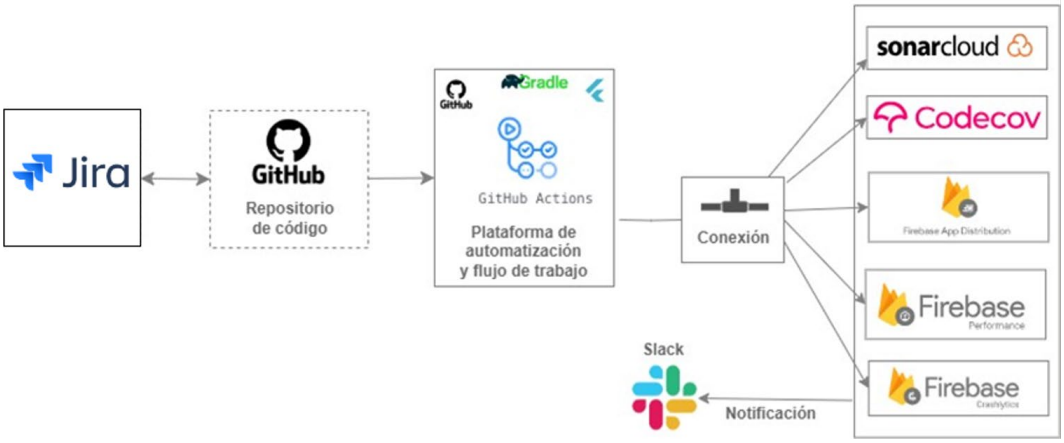

Figura 9. Pipeline del Esquema Mobile DevOps para planificar y construir FESTUM. Fuente: elaboración propia.

La Figura 9 muestra el flujo que sigue el pipeline. Este proceso inicia desde la planificación o gestión del proyecto donde en la herramienta Jira se escriben las épicas, historias, tareas que permiten esquematizar artefactos como product backlog, backlog detallado de producto y Sprints necesarios para construir FESTUM. En Jira se seleccionan tareas que son codificadas o programadas en un lenguaje y tecnología de programación, se realiza un commit y push en la rama master desde el entorno de desarrollo, en este caso Visual Studio Code (VSCode), hacia el repositorio en GitHub, haciendo uso de los comandos

proporcionados por Git. Estos comandos de Git posibilitan la transferencia de los cambios al repositorio GitHub. Una vez que estos cambios han sido subidos exitosamente inicia la ejecución del pipeline, con los distintos procesos establecidos en GitHub Actions. Un resumen de la ejecución del proceso es 1) Inicia el pipeline de CI/CD cada vez que se haga cambios en el repositorio. 2) Se ejecuta el trabajo (job) en una máquina virtual. 3) Verifica que todo el código necesario esté disponible para su procesamiento. 4) Configura Flutter en el entorno de ejecución utilizando una acción que establece la versión de Flutter adecuada. 5) Instala todas las dependencias del proyecto. 6) Realiza un análisis del código para identificar problemas en el código. 7) Ejecuta las pruebas automatizadas, para asegurar que el código funcione correctamente y que no se hayan introducido errores durante el desarrollo. 7) Construye el apk de la aplicación. 8) Se despliega la aplicación y lo distribuye. 9) Envía notificación sobre el progreso al canal de Slack. Una vez que la notificación se envía al canal de Slack, la secuencia de acciones se completa, a menos que surjan errores en alguna de las etapas que se están llevando a cabo. Si se produce un error en alguna fase, la secuencia generará un mensaje de error y no se finalizará, ya que cada proceso depende del éxito de sus pasos anteriores. Para llevar a cabo la construcción de la aplicación FESTUM, se utiliza la arquitectura de microservicios, por ello para el análisis, se usa mapa de capacidades. Un mapa de capacidades es una herramienta que describe y organiza las capacidades claves dentro de una organización para identificar y hacer cumplir objetivos concretos de la misma. En este caso permite delimitar y estructurar todos los procesos y funcionalidades que la aplicación requiere para que cumpla con las necesidades de los usuarios (Ver Figura 10).

Figura 10. Mapa de Capacidades usado para la construcción de la aplicación móvil FESTUM. Fuente: elaboración propia.

El Mapa de Capacidades como herramienta clave dentro del marco de la Arquitectura Empresarial, representa de forma estructurada las habilidades organizacionales necesarias para alcanzar los objetivos estratégicos [23]. En el caso de FESTUM, se utiliza el Mapa de

Capacidades para identificar y representar la cadena de valor y las capacidades que permitan el diseño e implementación de la propuesta de valor. El mapa de capacidades al organizarse en tres niveles jerárquicos permite representar en la cima las capacidades estratégicas (Nivel 1), que agrupan los dominios de alto nivel sobre los que se construye el modelo de negocio de FESTUM, estas capacidades definen de manera general y estable qué es lo que la organización realiza. En el Nivel 2 se encuentran las capacidades tácticas, que descomponen esas competencias estratégicas en servicios clave, estructurando así las funciones esenciales del negocio. Finalmente, en la base se ubican las capacidades operativas (Nivel 3), que detallan los elementos funcionales requeridos para poner en práctica cada capacidad táctica, conectándolas directamente con los procesos internos y la infraestructura tecnológica que las soporta.

El mapa de capacidades para FESTUM ofrece una visión clara y contextualizada del funcionamiento general de la aplicación móvil para lo cual se identificaron aspectos clave del ciclo de vida de un evento entre los que constan su creación, asistencia y ejecución (Ver Figura 10). En cada fase del ciclo de vida, se identifica como generar valor para los usuarios como para los organizadores, priorizando elementos como la facilidad de uso, la compatibilidad entre sistemas y la eficiencia operativa.

Para el diseño estructural de la aplicación móvil, se utiliza el patrón de diseño Domain-Driven Design (DDD), con el objetivo de construir una arquitectura limpia, modular y escalable. Este enfoque se basa en modelar el dominio del problema, alineando el software con las reglas de negocio y con las dinámicas específicas del entorno. Para ello se definen interfaces de repositorios que permiten abstraer el acceso a los datos, manteniendo separada la lógica de negocio de los aspectos técnicos relacionados con la infraestructura, estas implementaciones de los repositorios delegan las tareas de persistencia a los datasources, reforzando la independencia entre capas. Adicional a ello, se utilizaron servicios de dominio para encapsular lógica de las entidades, lo que permite llevar a cabo operaciones simples como complejas que involucran varios componentes del sistema. Finalmente, se incorporaron eventos de dominio para garantizar que los módulos de la solución de software reaccionen de forma adecuada y mantengan la consistencia interna del sistema. En la Figura 11 se visualiza el comportamiento del patrón DDD (domain driven design) donde la comunicación fluye desde la UI (interfaz de usuario) sobre la cual se efectúa la interacción con los usuarios y con el sistema. Los controladores de la UI capturan acciones del usuario y envían solicitudes a los servicios de aplicación, quienes se encargan de actuar como intermediarios, coordinando y gestionando el flujo de datos y operaciones de negocio entre la interfaz de usuario y la lógica del dominio. La orquestación de las operaciones del dominio con entidades y servicios de dominio que constituyen el núcleo del sistema, encapsulan la lógica y reglas del negocio a través de entidades para el acceso a los datos que utilizan infraestructura que permite la gestión de datos o datasources para permitir el intercambio de datos, finalmente, los datos se transforman en DTO's (Data Transfer Objects) y se presentan en la UI.

Para el desarrollo de la aplicación móvil FESTUM se utiliza Flutter [24] junto con el lenguaje de programación Dart con el objetivo de configurar un entorno de desarrollo que incluya las dependencias y herramientas para construir una solución multiplataforma a partir de una única base de código. El diseño de la interfaz de usuario (UI) de FESTUM es importante ya que se incluyen características y principios de diseño que permitan una experiencia visual coherente y adaptable, para ello se crean componentes reutilizables como botones, formularios y listas que agilizaron el proceso de codificación y facilitaron el mantenimiento de la aplicación, así como se configuran los widgets que ofrece Flutter y configuraciones adicionales para crear una interfaz responsiva, capaz de ajustarse fluidamente a distintos tamaños de pantalla y dispositivos.

Para la gestión del estado de los datos que se genera a través de la aplicación se utiliza Riverpod [25], un paquete de Flutter que facilita la gestión de estados reactivos y la inyección de dependencias. Esto permite una gestión del estado eficiente y un flujo de datos claro entre la UI y la lógica de negocio. Además, la integración con Firebase [26] es otro componente que se incluye en el desarrollo, especialmente Firebase Authentication

con el objetivo de gestionar el registro e inicio de sesión de los usuarios, ofreciendo soporte para múltiples métodos de autenticación, incluyendo correo electrónico y redes sociales, además se usa Firestore para crear una base de datos en tiempo real que permite el almacenamiento y sincronización de datos de eventos, usuarios y otras entidades propias de FESTUM. Finalmente, se utiliza Firebase Storage para almacenar archivos multimedia, como imágenes y videos, asociados con los eventos y usuarios y para asegurar los recursos backend contra accesos no autorizados, se utiliza App Check en Firebase, verificando que las solicitudes provengan de la aplicación autenticada y asegurando que solo las solicitudes legítimas puedan interactuar con los servicios backend.

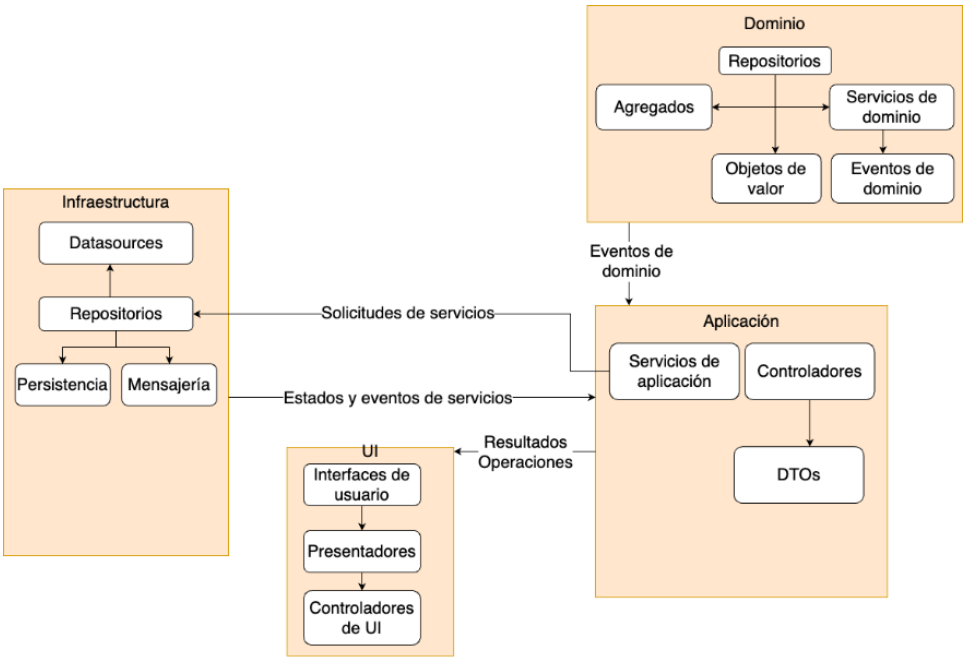

Figura 11. Patrón de diseño DDD (Domain Driven Design). Fuente: elaboración propia.

3.1.6 Fase medir

Posterior a la construcción de la aplicación móvil FESTUM se realiza un conjunto de pruebas que permitan evaluar el rendimiento en un entorno real, para ello 30 usuarios utilizan la aplicación durante un evento diseñado específicamente para este propósito. A lo largo de la jornada del evento, los asistentes interactuaron con la aplicación para realizar diversas acciones entre las que destacan el registrar su asistencia, explorar el contenido del evento y proponer sugerencias musicales mediante la integración con la API de Spotify. Este entorno controlado permitió observar la interacción de los usuarios con la aplicación y recolectar datos que permitan evaluaar funcionalidad y rendimiento. En la Figura 12 se muestra la implementación del API de Spotify [27] dentro de FESTUM con la cual se puede sugerir canciones a los usuarios durante el evento donde se recopilaron métricas sobre la cantidad de solicitudes realizadas al API, las canciones sugeridas y la satisfacción del usuario con las recomendaciones musicales.

La métrica que permite visualizar las solicitudes realizadas a través de FEStum muestra el recuento de solicitudes al API de Spotify durante el evento (Figura 13). Se observa que las solicitudes, marcadas con color azul, son exitosas respecto de las solicitudes con errores las cuales se marcan con color verde. En la figura se muestra la actividad de la aplicación en tiempo real y la interacción de los usuarios con la función de recomendación musical incluida dentro de FESTUM. Los picos de solicitudes corresponden a momentos en los que varios usuarios solicitaron recomendaciones simultáneamente, registrando un máximo de aproximadamente 17 a 18 solicitudes por minuto.

Figura 12. Arquitectura de funcionamiento de Festum con API de Spotify. Fuente: elaboración propia.

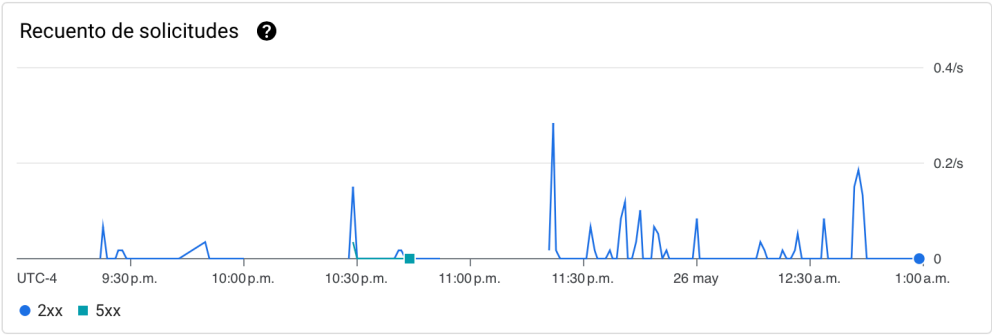

Figura 13. Métrica de solicitudes del API de Spotify. Fuente: elaboración propia.

3.1.7 Fase de lanzamiento

En la Figura 14, se presenta el diseño de la arquitectura del sistema que se usa para la implementación y lanzamiento de FESTUM. Esta arquitectura se divide en dos partes principales que son lado del cliente y lado del servidor: En el lado del cliente (frontend), se cuenta el framework Flutter que permitirá la representación visual en aplicaciones Android. En el lado del servidor (backend), se distinguen componentes no funcionales que respaldan la arquitectura de microservicios, tales como, Eureka, API Gateway. Además, se desarrollarán componentes funcionales, que corresponden a los microservicios de productos (asociado a lo que ofrece FESTUM), cliente (los usuarios con rol administrados, invitado), pedido (peticiones, solicitudes o actividades), que serán desarrollados utilizando SpringBoot [28]-[31]. Cada uno de estos microservicios tendrá acceso exclusivo a la base de datos relacional (MySQL) y base de datos no relacional (Firebase Cloud Firestore). Para gestionar eficazmente estos microservicios y tenerlos contenerizados, se utilizará Docker Host. La seguridad de acceso a estos servicios se garantizará mediante JWT. Finalmente, para gestionar los eventos a través de la mensajería, se utilizará Kafka. Es importante recalcar que se usa Spring WebFlux, para la base de datos NoSQL.

Para la construcción de FESTUM en un entorno móvil se usa Spring Boot y Spring WebFlux que permite programación reactiva con lo que se puede lograr escalabilidad y mejorar el rendimiento con baja latencia, además se utiliza Spring Data R2DBC, también conocido como Reactive Relational Database Connectivity, que representa una especificación diseñada para incorporar bases de datos SQL a través de controladores reactivos, esta tecnología simplifica el desarrollo de aplicaciones Spring reactivas que emplean tecnologías de acceso a datos relacionales. A través de la programación reactiva y el enfoque sin bloqueo, se mejora la eficiencia de la aplicación ya que se limita el uso de recursos y con ello se logra escalabilidad de la aplicación FESTUM.

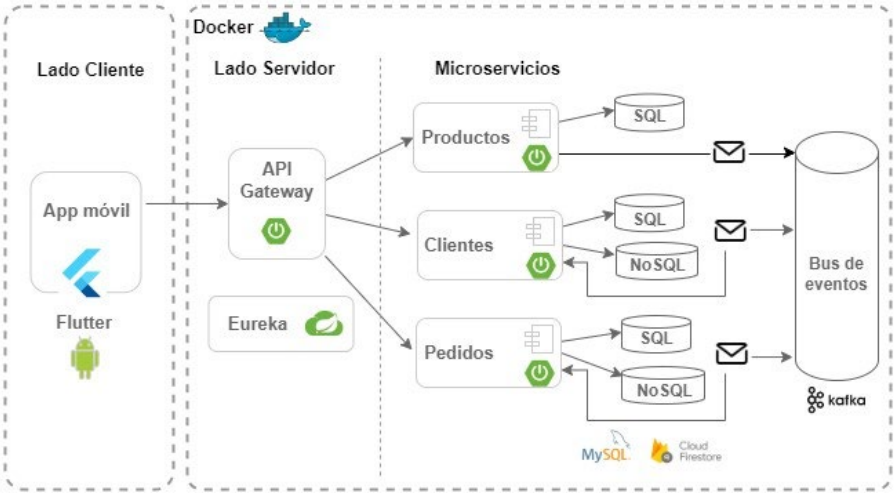

Figura 14. Diagrama del diseño de la arquitectura del sistema. Fuente: elaboración propia.

4. RESULTADOS Y DISCUSIÓN

En esta sección se evidencia el valor agregado de la investigación, donde la sinergia entre Design Thinking, Lean Startup y Scrum ha permitido implementar en un caso práctico FESTUM. La integración de marcos ágiles y herramientas en conjunto dentro del desarrollo ágil de software ha sido poco explorada, es por ello que, la investigación aporta una ruta clara desde la ideación hasta la implementación técnica. La aplicación móvil FESTUM, que es el resultado de la implementación de la metodología, ha sido validada desde el punto de vista de experiencia y funcionalidad con usuarios reales a través de herramientas como Maze y Mentimeter, y aunque el tamaño de la muestra de usuarios puede ser una limitación para generalizar el resultado a gran escala, es una oportunidad para aplicar la metodología a otro tipo de aplicaciones o contextos. Desde el punto de vista metodológico y técnico, FESTUM se construye bajo una arquitectura escalable donde el uso de microservicios, contenedores Docker y despliegue en Google Cloud con patrones como Domain-Driven Design (DDD), demuestra una alineación con las tendencias actuales en ingeniería de software. Para dar soporte a esta sección y una vez expuesto en resumen el valor agregado de nuestro trabajo y algunas implicaciones prácticas, es necesario dar respuestas a las tres preguntas de investigación.

RQ1. ¿Cómo se integran los frameworks para estructurar el desarrollo de software de manera ágil y eficiente?

Design Thinking, Lean Startup y Scrum proporcionan un conjunto de fases, actividades y herramientas que permiten la implementación de técnicas específicas y mejoran la eficiencia del equipo de trabajo para el análisis, diseño y construcciones de soluciones o productos de software. Para el caso de estudio FESTUM, las fases de Empatizar, Definir e Idear de Design Thinking permitieron identificar el problema, las necesidades, los usuarios para los cuales se busca una propuesta de solución creativa e innovadora a través de la creación de los mapas de propuestas de valor, perfiles de clientes y la generación de ideas mediante brainstorming y SCAMPER. Para las fases de Prototipar y Testear se utiliza Figma como herramienta que permitió crear wireframes y mockups en base a un diseño de interfaz de usuario que se puede validar con usuarios, garantizando que los diseños sean mejorados en base a la retroalimentación que se reciba por parte de los usuarios.

En el contexto de Lean Startup, herramientas como Mentimeter y Maze se utilizan en las fases de Construir, Medir y Aprender. Mentimeter que se usa en la fase de aplicación de encuestas para evaluar usabilidad a través de heurísticas de Nielsen que permiten garantizar que los productos mínimos viables (PMV) cumplan con los estándares de usabilidad. Maze como herramienta facilita la recopilación de datos y el análisis de la experiencia del usuario, lo que permite medir efectivamente el rendimiento del PMV y con

los resultados realizar ajustes continuos en los prototipos. En Scrum, herramientas como Jira en la fase de Planificación son usados para gestionar el backlog y organizar los sprints, mientras que una combinación de tecnologías de desarrollo (Flutter, Dart, Firebase, NestJS) y herramientas de pruebas (Firebase Test Lab) aseguran un desarrollo robusto y pruebas efectivas. Finalmente, Docker y Google Cloud son usadas para el lanzamiento de FESTUM, proporcionando una solución desplegada en contenedores que sea escalable y eficiente. La integración de estas herramientas en cada fase de los frameworks ágiles optimiza el flujo de trabajo y garantiza la calidad y funcionalidad de FESTUM.

RQ2. ¿Cuáles son las fases comunes entre Design Thinking, Lean Startup y Scrum que facilitan la evolución de un MVP hacia una arquitectura funcional basada en microservicios?

Design Thinking, Lean Startup y Scrum como metodologías tienen enfoques particulares cada uno, sin embargo, comparten algunas fases clave que facilitan la evolución de un PMV hacia una aplicación funcional que puede ser implementada a través de arquitectura monolítica o de microservicios, asegurando escalabilidad, modularidad y adaptabilidad. El proceso de descubrimiento y definición del problema, presente en Design Thinking (Empatizar, Definir), Lean Startup (Ajuste al Problema/Solución) y Scrum (Refinamiento del Product Backlog), establece las bases para comprender las necesidades del usuario y definir los requisitos iniciales. Esta fase es importante en la adopción de una arquitectura de microservicios, ya que permite identificar dominios independientes y la segmentación de funcionalidades en módulos autónomos. Luego en la fase de ideación y diseño de la solución, correspondiente a Design Thinking (Idear), Lean Startup (MVP Canvas, Experimentación) y Scrum (Sprint Planning), se realiza la concepción de soluciones modulares, donde el PMV es diseñado con principios de desacoplamiento para facilitar su evolución hacia microservicios. En esta etapa, se identifican los bounded contexts y las relaciones entre servicios, alineando el diseño de software con los principios de Domain-Driven Design (DDD).

La siguiente fase que corresponde al desarrollo iterativo y validación, es abordada en Design Thinking (Prototipado y Evaluación), Lean Startup (Build-Measure-Learn) y Scrum (Desarrollo en Sprints). En esta fase, la implementación del PMV sigue un enfoque incremental, donde cada iteración valida hipótesis y ajusta los requerimientos basados en métricas de desempeño y feedback del usuario. Este proceso de experimentación guiada minimiza el riesgo de crear un sistema monolítico al permitir que las funcionalidades sean implementadas como servicios independientes desde el inicio. Adicionalmente, la fase de optimización y escalabilidad, presente en Lean Startup y Scrum (Sprint Retrospective & Continuous Deployment), permite la mejora continua de los servicios mediante prácticas como la observabilidad distribuida, balanceo de carga y optimización del consumo de recursos. Finalmente, la fase de despliegue implica la automatización del ciclo de vida del software mediante la integración y entrega continua, contenedorización con Docker, y orquestación de microservicios con herramientas como API Gateway. Así, la convergencia de estos enfoques garantiza una transición estructurada desde un PMV hacia una arquitectura escalable, resiliente y alineada con los principios de agilidad y software distribuido.

RQ3. ¿Cómo el despliegue mediante contenedores garantiza la escalabilidad y el rendimiento de la aplicación móvil?

El despliegue de aplicaciones de software en contenedores ofrece importantes ventajas en términos de escalabilidad, rendimiento y seguridad debido al entorno de ejecución estandarizado, ligero y eficiente dependiendo del proveedor Cloud que se utilice. En un entorno dockerizado por ejemplo se puede encapsular todas las dependencias necesarias para ejecutar la aplicación, lo que reduce problemas tales como compatibilidad y garantiza un comportamiento consistente de la aplicación en distintos entornos. Desde el punto de vista de escalabilidad, los contenedores facilitan la adopción de arquitecturas

basadas en microservicios, lo que permite distribuir la carga de la aplicación de manera eficiente y escalar horizontalmente bajo demanda. El rendimiento como atributo de calidad, en contenedores mejora significativamente la eficiencia del sistema, ya que operan directamente sobre el núcleo del sistema operativo, sin la sobrecarga que suele existir en máquinas virtuales, por ejemplo. Tecnologías complementarias como Eureka permiten balancear la carga entre múltiples instancias del backend, asegurando una distribución equilibrada de las solicitudes, además se logra una gestión optimizada del caching y del almacenamiento persistente, lo que reduce la latencia y mejora el tiempo de respuesta de la aplicación, finalmente se puede mencionar que la aplicación móvil, en este caso FESTUM, ha sido desarrollada usando Mobile DevOps lo permite realizar despliegues rápidos, seguros y con actualizaciones incrementales sin interrumpir el servicio.

Una vez que se ha dado respuesta a las preguntas de investigación del presente trabajo, se presentan los resultados de las encuestas realizadas a los asistentes del evento que participaron en la prueba cerrada de la aplicación FESTUM (Ver Tabla 1). Las encuestas diseñadas se usaron para evaluar las funcionalidades de la aplicación y la satisfacción de los usuarios que utilizaron la aplicación. En resumen, se muestran los principales hallazgos, entre los que destacan que 23 asistentes completaron la encuesta, proporcionando una base sólida para el análisis de los resultados. El análisis sobre las preguntas evidencia que la mayoría de los usuarios (22 de 23) consideraron que la aplicación mejoró su experiencia en la creación y asistencia de eventos, lo que valida la propuesta de valor inicial de la aplicación. La intención de uso futuro es alta, con 14 de 16 usuarios afirmando que volverían a usar la aplicación, lo que indica una fuerte aceptación entre los usuarios. La funcionalidad de compartir fotos fue bien aceptada por los usuarios, aunque un número significativo no intentó utilizar esta función debido a la baja conectividad de la infraestructura dentro del escenario de prueba.

Tabla 1. Tabulación de experiencias de usuario durante la fase de testing. Fuente: elaboración propia.

Pregunta	Respuesta	Cantidad
¿Consideras que festum mejoró tu experiencia en la creación y asistencia de eventos?	Sí, significativamente	13
	Sí, en cierta medida	9
	Neutral	1
	No, no mejoró mucho	0
	No, no mejoró en absoluto	0
¿Volverías a usar FESTUM?	Sin duda alguna	14
	Tal vez	2
	Me lo pensaría	0
	No lo haría	0
¿Pudiste compartir fotos durante el evento en tiempo real sin problemas?	Sí, sin problemas	8
	Sí, pero con algunos problemas	0
	No lo intenté	7
	No, tuve problemas	0
¿Sugeriste música en vivo durante el evento?	Sí, varias veces	6
	Sí, algunas veces	4
	No sugerí	4
	Sí, sin duda alguna	8
	Sí, pero hay cosas que mejorar	6
	No	0
¿Cómo calificarías la experiencia de encontrar eventos en festum?	Excelente	6
	Buena	7
	Neutral	1
	Mala	0
	Muy mala	0
En una escala del 1 al 10, ¿Cómo calificarías tu experiencia general con festum?	1-3	0
	4-7	0

	8-10	14

La función de sugerir música de FESTUM fue utilizada por la mayoría de los usuarios, lo que muestra su popularidad y relevancia durante el evento. Aunque la funcionalidad de sugerir música fue bien recibida, algunos usuarios identificaron áreas de mejora, lo que ofrece oportunidades para optimizar esta característica. La mayoría de los usuarios calificó positivamente la experiencia de encontrar eventos, lo que refuerza la efectividad de la aplicación en este aspecto. Los usuarios calificaron su experiencia con la aplicación FESTUM entre 8 y 10, lo cual es un indicador con un alto nivel de satisfacción a nivel general de la aplicación. Los usuarios proporcionaron comentarios abiertos, destacando aspectos positivos tales como la facilidad de uso y la funcionalidad de las sugerencias musicales, así como sugerencias para mejorar ciertas características a nivel de diseño de la UI. Los resultados de las encuestas muestran un nivel de satisfacción intermedio – alto con la aplicación y confirman que la mayoría de las funcionalidades fueron aceptables. La funcionalidad de sugerir música a través del API de Spotify fue particularmente popular, aunque algunos usuarios mencionaron que aún se deben hacer mejoras. La experiencia de encontrar eventos y la intención de uso futuro también tuvieron una escala de valor importante. Estos hallazgos validan las ideas planteadas al inicio del desarrollo de la aplicación, confirmando que las herramientas y metodologías utilizadas, como Design Thinking y Lean Startup, resultaron efectivas para crear una solución que satisface las necesidades de los usuarios.

5. CONCLUSIONES

La integración de frameworks ágiles como Design Thinking, Lean Startup y Scrum en el desarrollo de la aplicación móvil FESTUM ha demostrado ser una estrategia efectiva para la creación de soluciones empresariales innovadoras y centradas en el usuario. Como resultado de la integración se puede evidenciar que el diseño no se trata solo de la creación del producto, sino de entender y mejorar la UI y las funcionalidades gracias a la retroalimentación que brindan las personas. Durante el desarrollo del presente trabajo se evidencia que a través de las metodologías se puede desarrollar soluciones más sostenibles y efectivas, involucrando a los stakeholders desde el inicio del proceso para asegurar que las soluciones sean prácticas y ampliamente aceptadas. La metodología Design Thinking permitió comprender las necesidades de los usuarios apoyado para ello en las fases de empatía, definición e ideación, utilizando herramientas como MindMeister, Miro y Jamboard. La creación de prototipos y su validación con Figma aseguraron que las ideas se convirtieran en soluciones viables y funcionales. Al utilizar estos enfoques colaborativos y centrados en el usuario se garantiza que el producto final sea funcional y esté alineado con las expectativas y deseos de los usuarios.

Herramientas como Mentimeter y Maze, dentro de las fases de construcción y medición de Lean Startup, permitieron la creación y evaluación del PMV de FESTUM, adicional a ello también permitió iterar rápidamente en función de la retroalimentación por parte de los usuarios. La implementación de Scrum en herramientas como Jira para la planificación y tecnologías como Flutter, Dart, Firebase, y NestJS para el desarrollo, aseguraron un proceso de desarrollo ágil y robusto. Para el despliegue de la solución integral Docker y Google Cloud proporcionan una infraestructura escalable y eficiente especialmente para cumplir con la sostenibilidad, la seguridad y el rendimiento. El uso de una arquitectura de microservicios en el desarrollo de la aplicación FESTUM permitió modularidad y escalabilidad de la aplicación, debido a que cada microservicio, desarrollado y desplegado de manera independiente, contribuyó a una mayor flexibilidad y facilidad de mantenimiento. La aplicación de patrones de diseño como Domain-Driven Design (DDD) y la integración con servicios como Firebase para la autenticación y almacenamiento de datos permitió establecer configuraciones enfocados en la robustez y seguridad de la aplicación. La prueba cerrada de la aplicación proporcionó datos valiosos que validan la propuesta de valor con lo que se destaca la satisfacción de los usuarios con las

funcionalidades de FESTUM. Finalmente se puede concluir que, en conjunto, la combinación de estos frameworks ágiles y la arquitectura de microservicios permitió optimizar el proceso de desarrollo y con ello se garantiza una solución final adaptable, escalable y altamente satisfactoria para los usuarios.

6. ACERCA DEL ARTÍCULO

Agradecimientos y financiación: Este trabajo fue realizado con el apoyo del Departamento de Ciencias de la Computación de la Universidad Técnica Particular de Loja.

Declaración del investigador: Declaro que los resultados presentados en este trabajo son originales, no han sido publicados previamente y cumplen con los principios de rigor científico y ética académica. Asimismo, asumo plena responsabilidad ante cualquier reclamación relacionada con derechos de propiedad intelectual.

Contribuciones de autoría:

José Luis Benitez Coronel: Aplicación de la metodología, construcción de la solución de software, redacción, revisión y edición.
Ronin David Carrión Carrión: Aplicación de la metodología, construcción de la solución de software, redacción, revisión y edición.
Fernanda Soto: Definición de la metodología y validación del diseño, redacción, revisión y edición.
Daniel Guamán: Definición de la metodología y validación del diseño, redacción, revisión y edición.

Conflictos de interés:

Los autores declaran no tener ningún conflicto de interés.

REFERENCIAS

[1] M. Zakrzewska, K. Piwowar-Sulej, S. Jarosz, A. Sagan, and M. Sołtysik, “The linkage between Agile project management and sustainable development: A theoretical and empirical view,” *Sustainable Development*, vol. 30, no. 5, pp. 855–869, Oct. 2022. https://doi.org/10.1002/sd.2285

[2] M. Dick, J. Drangmeister, E. Kern, and S. Naumann, “Green software engineering with agile methods,” in *Proc. 2nd Int. Workshop on Green and Sustainable Software (GREENS)*, San Francisco, CA, USA, May. 2013, pp. 78–85. https://doi.org/10.1109/GREENS.2013.6606425

[3] C. C. Venters *et al.*, “Software sustainability: Research and practice from a software architecture viewpoint,” *J. Syst. Softw.*, vol. 138, pp. 174–188, Apr. 2018. https://doi.org/10.1016/j.jss.2017.12.026

[4] L. Waidelich, A. Richter, B. Kölmel, and R. Bulander, “Design thinking process model review,” in *2018 IEEE Int. Conf. on Engineering, Technology and Innovation (ICE/ITMC)*, Stuttgart, Germany, Jun. 2018, pp. 1–9. URL

[5] Y. Mansoori, T. Karlsson, and M. Lundqvist, “The influence of the lean startup methodology on entrepreneur-coach relationships in the context of a startup accelerator,” *Technovation*, vol. 84–85, pp. 37–47, Jun.–Jul. 2019. https://doi.org/10.1016/j.technovation.2019.03.001

[6] K. Schwaber, and J. Sutherland, *Der visuelle scrum guide,* Accessed: Dec. 2, 2025. [Online]. Available: https://wiki.librescrum.org/der_visuelle_scrum_guide.pdf

[7] T. Brown, *Diseñar El cambio: Como el Design Thinking Transforma Organizaciones E Inspira la Innovacion*. Bogotá, Ediciones Urano, 2020. URL

[8] E. Ries, *The lean Startup: How today’s entrepreneurs use continuous innovation to create radically successful businesses*. Crown Business, 2011. https://ia800509.us.archive.org/7/items/TheLeanStartupErickRies/The%20Lean%20Startup%20-%20Erick%20Ries.pdf

[9] ScrumAlliance, "The 2020 Scrum Guide and Related Content," resources.scrumalliance.org. Accessed: Dec. 2, 2025. [Online]. Available: https://resources.scrumalliance.org/Article/2020-scrum-guide-related-content

[10] Festum, "Crea, Vive, Disfruta," festum.app. Accessed: Dec. 2, 2025. [Online]. Available: https://festum.app/

[11] L. De Lauretis, "From Monolithic Architecture to Microservices Architecture," in *2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, Berlin, Germany, 2019, pp. 93-96. https://ieeexplore.ieee.org/document/8990350

[12] S. Newman, *Building microservices: Designing fine-grained systems*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2021. https://www.oreilly.com/library/view/building-microservices-2nd/9781492034018/

[13] A. Ghezzi, and A. Cavallo, "Agile business model innovation in digital entrepreneurship: Lean Startup approaches," *J. Bus. Res.*, vol. 110, pp. 519–537, Mar. 2020. https://doi.org/10.1016/j.jbusres.2018.06.013

[14] M. Stocker, O. Zimmermann, U. Zdun, D. Lübke, and C. Pautasso, "Interface quality patterns: Communicating and improving the quality of microservices APIs," in *Proceedings of the 23rd European Conference on Pattern Languages of Programs*, New York, NY, USA, ACM, Digital Library, 2018, pp. 1–16. https://doi.org/10.1145/3282308.3282319

[15] C. Singh, N. S. Gaba, M. Kaur, and B. Kaur, "Comparison of Different CI/CD Tools Integrated with Cloud Platform," in *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, India, 2019, pp. 7-12. https://ieeexplore.ieee.org/document/8776985

[16] J. Kreisa, "DockerCon 2020: And … That's a Wrap," docker.com. Accessed: Dec. 2, 2025. [Online]. Available: https://www.docker.com/blog/dockercon-2020-and-thats-a-wrap/

[17] D. Taibi, V. Lenarduzzi, C. Pahl, and A. Janes, "Microservices in agile software development: A workshop-based study into issues, advantages, and disadvantages," in *Proceedings of the XP2017 Scientific Workshops*, Cologne, Germany, May. 2017, Article 23, pp. 1–5. https://doi.org/10.1145/3120459.3120483

[18] D. A. Barrios Contreras, "Arquitectura de microservicios," *Tecnol. Investig. Academia TIA*, vol. 6, no. 1, pp. 36–46, Mar. 2018. https://revistas.udistrital.edu.co/index.php/tia/article/view/9687/pdf

[19] M. Castillo-Vergara, A. Alvarez-Marin, and R. Cabana-Villca, "Design thinking: como guiar a estudiantes, emprendedores y empresarios en su aplicación," *Ingeniería Industrial,* vol. 35, no. 3, pp. 301-311, Sep.-Dec. 2014. https://www.redalyc.org/pdf/3604/360433598006.pdf

[20] S. Newman, "Build," in *Building microservices: Designing fine-grained systems*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2021. https://www.oreilly.com/library/view/building-microservices-2nd/9781492034018/

[21] S. De Santis, L. Florez, D. V. Nguyen, and E. Rosa, *Evolve the monolith to microservices with Java and Node*. IBM, Redbooks, 2016. [Online]. Available: https://www.redbooks.ibm.com/abstracts/sg248358.html

[22] AWS, "¿Qué es Docker?," aws.amazon, 2021. Accessed: Dec. 2, 2025. [Online]. Available: https://aws.amazon.com/es/docker/

[23] R. Harrison, *TOGAF™ 9 Foundation Study Guide Preparation for the TOGAF 9 part 1 Examination*, The Open Group, Van Haren, 2016. Accessed: Dec. 2, 2025. [Online]. Available: https://www.vanharen.store/samplefile/api/downloader/getId/978908753681C?srsltid=AfmBOooANYTen8xzoMyj7niMZ_p61zgPS1oUbD4IJJzvVXqmoxZPDyGk

[24] E. Windmill, *Flutter in action,* New York, NY, USA: Manning Publications, 2020. https://www.manning.com/books/flutter-in-action?manning_source=marketplace&manning_medium=catalog

[25] R. Rousselet, *Riverpod: Un Framework de Caché y Enlace de Datos Reactivo,* Riverpod, (2023), biblioteca (framework), Accessed: Dec. 2, 2025. [Online]. Available: https://riverpod.dev/es/

[26] J. Taplin, and A. Lee, *Firebase,* Google, (2013), startup Envolve, Accessed: Dec. 2, 2025. [Online]. Available: https://firebase.google.com/

[27] Spotify, "Spotify for Developers," developer.spotify, 2024. Accessed: Dec. 2, 2025. [Online]. Available: https://developer.spotify.com/documentation/web-api

[28] Spring, "Spring Framework Overview," spring.io, 2023. Accessed: Dec. 2, 2025. [Online]. Available: https://docs.spring.io/spring-framework/reference/overview.html

[29] A. Cadavid Navarro, J. D. Martínez Fernández, and J. Morales Vélez, " Revisión de metodologías ágiles para el desarrollo de software," *PROSPECTIVA,* vol. 11, no. 2, pp. 30-39, Jul.-Dec. 2013. https://www.redalyc.org/pdf/4962/496250736004.pdf

[30] J. Pourdehnad, D. Wilson, and E. Wexler, "Systems & design thinking: A conceptual framework for their integration," in *Proc. 55th Annual Meeting of the ISSS*, Hull, UK, 2011. https://journals.isss.org/index.php/proceedings55th/article/view/1650

[31] M. Stocker, O. Zimmermann, U. Zdun, D. Lübke, and C. Pautasso, "Interface quality patterns: Communicating and improving the quality of microservices APIs," in *Proceedings of the 23rd European Conference on Pattern Languages of Programs*, , New York, NY, USA, ACM, Digital Library, 2018, pp. 1–16. https://dl.acm.org/doi/10.1145/3282308.3282319