



DOCUMENTACIÓN TÉCNICA Y DE ARQUITECTURA:

# KAIROS RHYTHMOS

**Versión del Documento: 1.0 (MVP Espartano / KMP Release)**

**Rol:** Ingeniero de Software Mobile

**Autor:** Leonardo Ahumada

**Repositorio:** [leoahumadausa-Balam/KairosRhythmos\\_ShowCase](https://github.com/leoahumadausa-Balam/KairosRhythmos_ShowCase)

## Contenido

<b>1. ARQUITECTURA Y DISEÑO DE SOFTWARE</b>	2
1.1 Filosofía de Diseño: Clean Architecture sobre KMP	2
1.2 Modelo de Objetos (Dominio)	3
1.3 Stack Tecnológico y Decisiones de Ingeniería	3
<b>2. PLANTEAMIENTO DEL PROBLEMA Y SOLUCIÓN</b>	4
2.1 Situación Original (AS-IS)	4
2.2 Solución Implementada (TO-BE)	4
<b>3. SEGURIDAD Y PERSISTENCIA DE DATOS</b>	5
3.1 Almacenamiento Local (Sandboxing)	5
<b>4. FUNCIONALIDADES CLAVE Y UX (EXPERIENCIA DE USUARIO)</b>	5
4.1 Ingeniería Gráfica: El Reloj Trigonométrico	5
4.2 "Mini-Kairos": Sistema de Interrupción Crítica	6
4.3 Ingesta de Datos: Inscripción Contextual	7
<b>5. LÓGICA DE NEGOCIO AVANZADA</b>	8
5.1 Gestión de Estados Temporales	8
<b>6. INFRAESTRUCTURA Y FUTURO (ROADMAP V2.0)</b>	9
6.1 Automatización: Agente "MiniKairos" Inteligente	9
6.2 Visualización Inmersiva: Reloj Focus (Micro-Gestión)	9
<b>7. TRAZABILIDAD Y AUDITORÍA</b>	10

# 1. ARQUITECTURA Y DISEÑO DE SOFTWARE

## 1.1 Filosofía de Diseño: Clean Architecture sobre KMP

El núcleo de **Kairos Rhythmos** se basa en una arquitectura de **Separación de Responsabilidades Estricta** implementada sobre **Kotlin Multiplatform (KMP)**.

A diferencia de las aplicaciones móviles tradicionales acopladas al sistema operativo, este sistema implementa una división basada en Dominios Puros:

- **Independencia de UI (The Spartan Monolith):** El 95% de la lógica de negocio (Entidades, Casos de Uso, Reglas de Tiempo) reside en el módulo compartido commonMain. El sistema no sabe si está ejecutándose en Android o iOS hasta el momento de renderizar.
- **Gestión de Estado Reactiva:** Se utiliza el patrón **MVI (Model-View-Intent)** mediante StateFlow, garantizando un flujo de datos unidireccional y predecible.

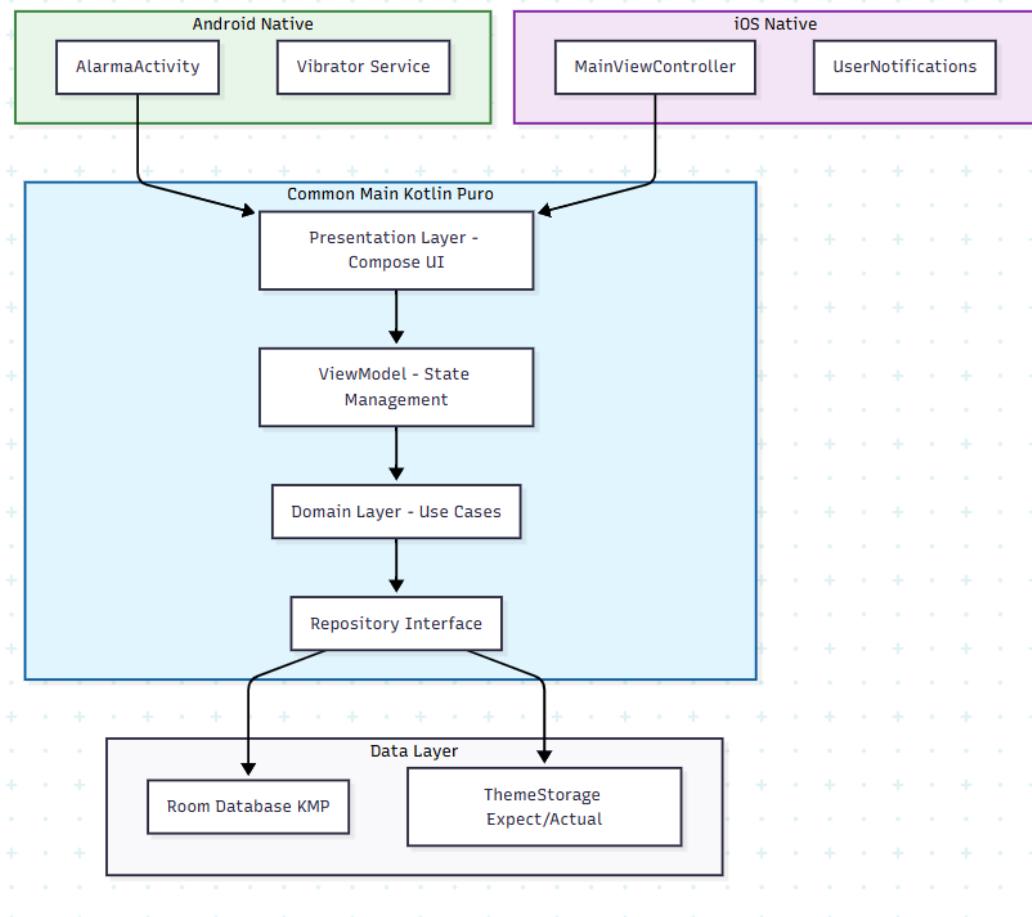


Diagrama de Arquitectura

## 1.2 Modelo de Objetos (Dominio)

El sistema modela el tiempo no como fechas estáticas, sino como **Flujos de Minutos**.

- **Entidad Tarea:** Encapsula datos y comportamiento de estado (PENDIENTE, EN\_CURSO, COMPLETADA, OMITIDA).
- **Entidad Ciclo:** Representa el día como un círculo finito (0 a 360 grados), eliminando la percepción de lista infinita.

## 1.3 Stack Tecnológico y Decisiones de Ingeniería

Capa / Componente	Tecnología	Justificación Técnica
Presentación (UI)	Compose Multiplatform	Permite una UI Declarativa Unificada. Se escribe una vez en Kotlin y se renderiza nativamente en Android (Canvas) e iOS (Skia), garantizando consistencia <i>pixel-perfect</i> .
Negocio (Core)	Kotlin 2.0 (Coroutines)	Lógica asíncrona no bloqueante. Uso de Flow y StateFlow para reactividad en tiempo real sin congelar la UI.
Persistencia	Room KMP (SQLite)	Base de datos relacional embebida multiplataforma. Permite que la UI reaccione automáticamente a cambios en los datos (Single Source of Truth).
Configuración	Expect/Actual Pattern	Implementación propia de ThemeStorage que mapea a <b>SharedPreferences</b> (Android) y <b>NSUserDefaults</b> (iOS) para máxima eficiencia sin librerías de terceros pesadas.
Hardware	Native Services	Acceso de bajo nivel (AlarmManager, VibrationEffect) injectado para garantizar interrupciones de alta prioridad ("Disciplina Espartana").

## 2. PLANTEAMIENTO DEL PROBLEMA Y SOLUCIÓN

### 2.1 Situación Original (AS-IS)

El usuario moderno sufre de una desconexión entre la planificación (Listas de Tareas digitales) y la ejecución física. Las apps actuales son "pasivas": envían notificaciones estándar que el usuario puede descartar con un simple *swipe*, fomentando la procrastinación y la acumulación de deuda técnica emocional.

### 2.2 Solución Implementada (TO-BE)

**Kairos Rhythmos** convierte el tiempo abstracto en magnitudes físicas (Arcos visuales, Masa y Vibración háptica).

- **Visibilidad Radial:** El usuario visualiza el día como un ciclo finito. Los huecos libres son tangibles.
- **Coerción de Atención (Mini-Kairos):** El sistema de alarmas rompe el flujo normal del dispositivo.
  - En Android, utiliza `FullScreenIntent` para sobreponerse a otras apps.
  - Implementa un patrón de vibración infinito (`Loop`) que exige interacción física.

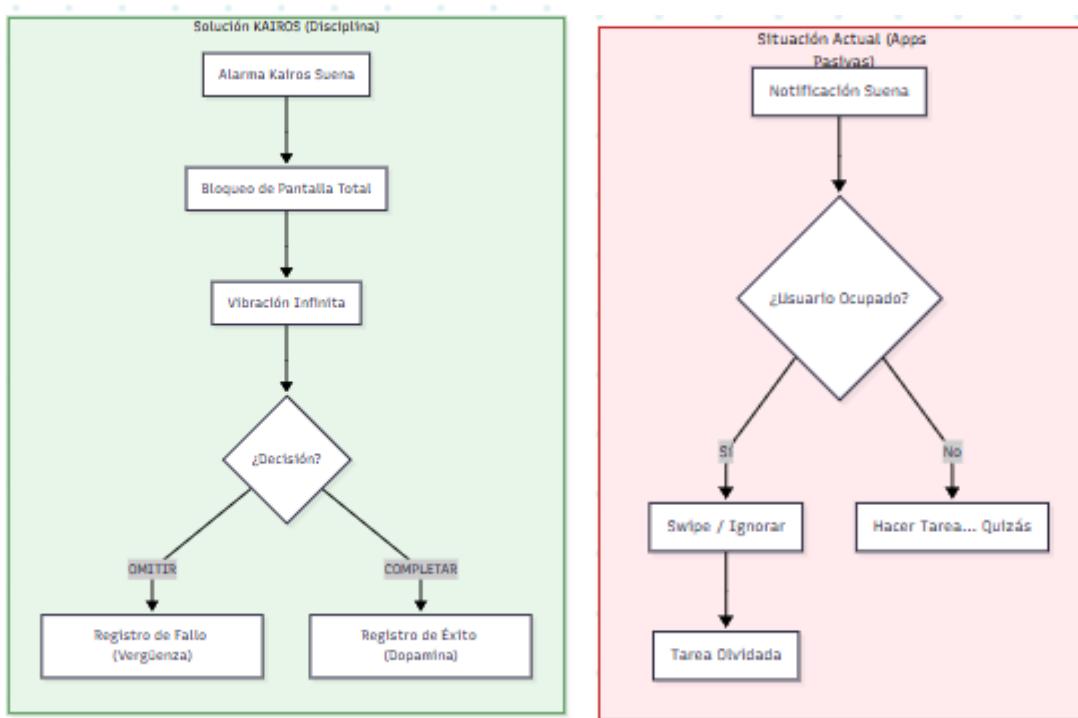


Diagrama de Flujo "AS-IS vs TO-BE" (Comparativa)

## 3. SEGURIDAD Y PERSISTENCIA DE DATOS

### 3.1 Almacenamiento Local (Sandboxing)

Aunque la aplicación no gestiona transacciones bancarias, los datos de productividad del usuario se tratan con integridad crítica.

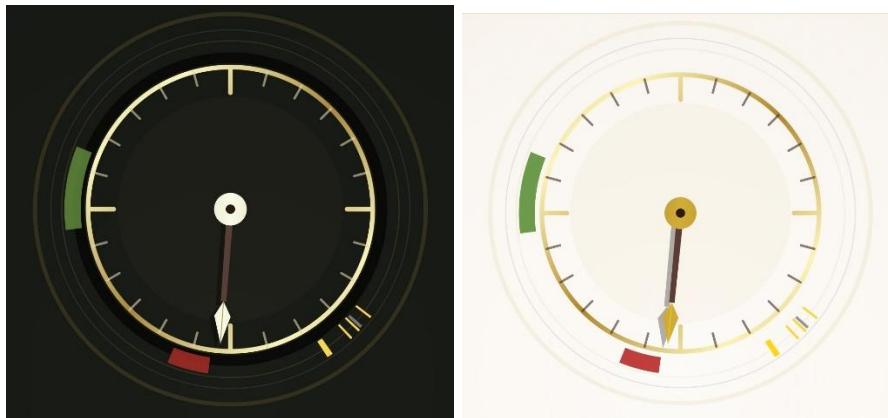
- **Aislamiento por SO:** La base de datos SQLite (Room) opera dentro del *sandbox* privado de la aplicación en cada sistema operativo, inaccesible para otras apps sin permisos root.
- **Transaccionalidad (ACID):** Las operaciones de escritura (Crear Tarea, Completar Tarea) son atómicas. Si falla la escritura en el historial, se revierte el estado visual, evitando inconsistencias de datos.

## 4. FUNCIONALIDADES CLAVE Y UX (EXPERIENCIA DE USUARIO)

### 4.1 Ingeniería Gráfica: El Reloj Trigonométrico

En lugar de utilizar *assets* de imagen estáticos, el componente central RelojKairos renderiza el tiempo matemáticamente en tiempo real utilizando el Canvas de Compose.

- **Cálculo Polar:** Convierte minutos del día (0-1440) en coordenadas angulares (0-360°) mediante funciones trigonométricas ( $\sin$ ,  $\cos$ ).
- **Dinámica Orbital:** Las tareas completadas no desaparecen; "migran" visualmente a una órbita exterior (`radiusEventosOuter`), creando un registro visual histórico del esfuerzo diario.



Pantalla del "Reloj Kairos" tema Oscuro y tema Claro

## 4.2 "Mini-Kairos": Sistema de Interrupción Crítica

Diseñado para combatir la "ceguera a las notificaciones", el sistema despliega el "**Informe de Batalla**" en el momento exacto en que finaliza un bloque de tiempo.

- **Loop Háptico:** Un patrón de vibración continua que no cesa hasta que el usuario interactúa.
- **Semántica de Crisis:** El diálogo bloquea la interacción con el resto del teléfono. Obsérvese en la captura que **no existe el botón "X" de cerrar ni la opción "Posponer 5 minutos"**. El usuario se enfrenta a una decisión binaria estricta: SÍ (COMPLETADA) o NO (OMITIR/FALLIDA), forzando la honestidad inmediata.



Pantalla de la Alarma

### 4.3 Ingesta de Datos: Inscripción Contextual

La creación de tareas no es un formulario ciego; es una inserción quirúrgica en el tiempo disponible.

- **Visualización Lineal de Disponibilidad:** En la parte superior del modal, el sistema "desenrolla" el reloj circular en una barra lineal (00:00 - 24:00). Esto permite al usuario ver instantáneamente los "huecos" libres (espacio crema) y los conflictos con tareas existentes (bloques de color) antes de guardar.
- **Taxonomía por Categorías:** El selector inferior permite asignar la "Energía" de la tarea (Productividad vs Descanso). Esta elección determinará el color del arco en el reloj principal, facilitando la lectura rápida del día.



Modal para ingreso de tareas

## 5. LÓGICA DE NEGOCIO AVANZADA

### 5.1 Gestión de Estados Temporales

Para optimizar el consumo de batería, el sistema no consulta la base de datos cada segundo.

- **Proyección de Tiempo:** El sistema calcula el estado de una tarea (PENDIENTE vs PASADA) proyectando el `minutoActual` del sistema contra los intervalos de las tareas cargadas en memoria.
- **Renderizado Eficiente:** La UI solo se redibuja cuando el minuto cambia, minimizando el uso de CPU y GPU.

### Eventos del Día

Estudiar kotlin12:30 • 60 min • EN CURSO

Partido de Padel17:30 • 120 min • PENDIENTE

08:28 • 5 min • COMPLETADA

08:45 • 6 min • OMITIDA

### Listado de tareas

## 6. INFRAESTRUCTURA Y FUTURO (ROADMAP V2.0)

### 6.1 Automatización: Agente "MiniKairos" Inteligente

La arquitectura está preparada para la integración de un módulo de Inteligencia Artificial Local (On-Device).

- **Objetivo:** Analizar los "huecos" geométricos en el anillo del reloj.
- **Funcionalidad:** Sugerir bloques de productividad óptimos basándose en el historial de cumplimiento del usuario (guardado en Room), transformando la app de una herramienta reactiva a un asistente proactivo.

### 6.2 Visualización Inmersiva: Reloj Focus (Micro-Gestión)

Para la fase de ejecución, se implementará una transición de contexto visual denominada "**Modo Arena**".

- **Túnel de Atención:** Al iniciar una tarea, el "Reloj Kairos" (que muestra las 24 horas) rota 180 grados visualmente para mostrar la "Cara Trasera".
- **Centralización en la Actividad:** En este modo, el anillo de tiempo ya no representa el día completo, sino exclusivamente la **duración de la tarea actual**.
  - Se elimina todo el "ruido visual" de las tareas futuras o pasadas.
  - Se amplifica la granularidad del tiempo, permitiendo visualizar el progreso minuto a minuto de la actividad presente, facilitando el estado de *Flow*.



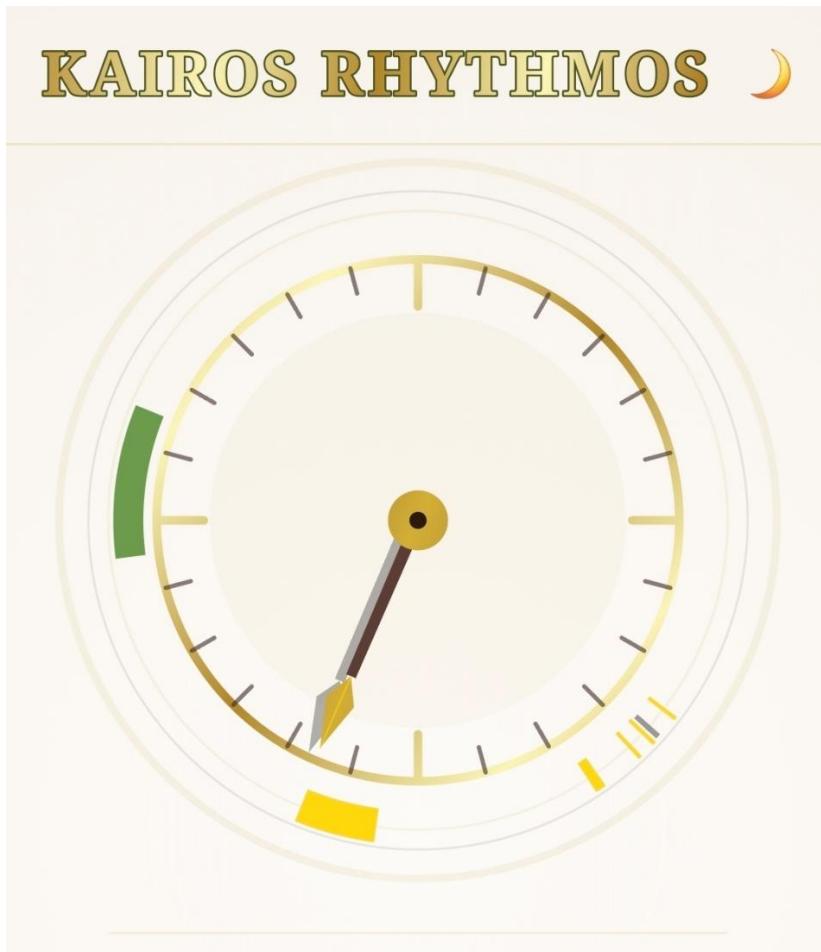
Vista del modo Focus y barra de progresión de la tarea

## 7. TRAZABILIDAD Y AUDITORÍA

### 7.1 Modelo de Verdad Única (SSOT)

El sistema implementa una filosofía de "**Honestidad Radical**". A diferencia de las listas de tareas tradicionales que permiten posponer indefinidamente, Kairos registra la realidad de la ejecución.

- **Inmutabilidad del Fallo:** Al sonar la alarma de término, la opción de "Reprogramar" se deshabilita intencionalmente en la base de datos local. El usuario debe marcar la tarea como NO COMPLETADA si no la realizó, generando un registro de auditoría honesto.
- **Auditoría Visual:** La "Cara Exterior" del reloj permite visualizar las discrepancias entre lo planificado (Anillo Interior) y lo ejecutado (Anillo Exterior), proveyendo métricas claras de autodisciplina.



**Mecanismo de Auditoría Visual**

## 7.2 Codificación Visual del Estado (Semántica de Colores)

El Reloj Kairos transforma el tiempo abstracto en un historial de rendimiento tangible mediante un sistema de tres estados visuales que actúan como un *Dashboard* en tiempo real:

1. **El Rastro del Éxito (Oro):** Las líneas doradas en la órbita exterior representan la disciplina consolidada. Cada arco es una tarea completada a tiempo, "calcificando" el esfuerzo productivo en el historial del día.
2. **La Huella de la Omisión (Gris/Rojo):** Los segmentos apagados denotan "tiempo muerto" o fallos en la voluntad. Son tareas ignoradas o canceladas que permanecen visibles para fomentar la responsabilidad (*accountability*) sin posibilidad de borrado.
3. **La Proyección del Futuro (Multicolor):** El anillo interior anticipa las actividades pendientes. Cada color corresponde a un subgrupo semántico (ej: **Azul** para *Descanso*, **Verde** para *Bienestar*), permitiendo al usuario preparar mentalmente el tipo de energía requerida para el siguiente bloque.