



UNIVERSIDAD DEL VALLE

FACULTAD DE INGENIERÍA

Codelab SEGUIMIENTO Y MÉTRICAS DEL PROYECTO

AUTORES:

Manuel Alexander Serna Jaraba - 202259345

Santiago Villa Salazar - 202259527

Adrian Felipe Velasquez Arias - 202259456

Edgar Fabian Rueda Colonia - 2259606

DOCENTE:

SALAZAR VICTORIA, Alvaro Hernando

CURSO:

Introducción a la Gestión de Proyectos de Software

Tuluá, Valle del Cauca

Noviembre de 2025



UNIVERSIDAD DEL VALLE

FACULTAD DE INGENIERÍA

Codelab SEGUIMIENTO Y MÉTRICAS DEL PROYECTO

AUTORES:

Manuel Alexander Serna Jaraba - 202259345

Santiago Villa Salazar - 202259527

Adrian Felipe Velasquez Arias - 202259456

Edgar Fabian Rueda Colonia - 2259606

DOCENTE:

SALAZAR VICTORIA, Alvaro Hernando

CURSO:

Introducción a la Gestión de Proyectos de Software

Tuluá, Valle del Cauca

Noviembre de 2025

1. Línea Base y Fuentes de Datos

Objetivo: Definir qué vamos a medir y de dónde sacamos los datos .

- Alcance (WBS): Basado en los requisitos previos (RF-01 a RF-10).
 - Paquetes de trabajo: Autenticación, Catálogo, Carrito, Checkout, Perfil, Gestión Admin.
- Presupuesto (BAC - Simulado): Para efectos del ejercicio EVM, asignaremos un valor monetario a las horas de desarrollo.
 - BAC (Budget at Completion): \$10,000 USD (Teórico: 5 devs x 100 horas x \$20/h).
- Fuentes de Datos:
 - Gestión de Trabajo: GitHub Projects / Jira (Estado de historias, Puntos de historia) .
 - Código y Calidad: SonarCloud (Deuda técnica, Bugs), Detekt/Ktlint (Estilo) .
 - DevOps/CI: GitHub Actions (Tiempos de build, frecuencia de despliegue) .
 - Operación: Firebase Console (Crashlytics, Uso de lecturas Firestore) .

2. EVM Operativo (Gestión del Valor Ganado)

Escenario Simulado: Estamos en la Semana 4 de un proyecto de 8 semanas. Ha habido problemas con la curva de aprendizaje de Jetpack Compose.

Tabla de Datos al Corte (Semana 4)

Métrica	Valor (\$)	Significado
BAC	\$10,000	Presupuesto total.
PV (Planned Value)	\$5,000	Deberíamos llevar el 50% del trabajo hecho.
EV (Earned Value)	\$4,000	Realmente hemos completado el 40% (Historias "Done").
AC (Actual Cost)	\$5,500	Hemos gastado más horas/recursos de lo planeado.

Cálculos de Desempeño

- Variación de Cronograma (SV): $EV - PV = 4,000 - 5,000 = -1,000\$$ (Retraso).
- Variación de Costo (CV): $EV - AC = 4,000 - 5,500 = -1,500\$$ (Sobrecosto).
- SPI (Índice de Cronograma): $EV / PV = 4,000 / 5,000 = 0.80$ (Vamos al 80% de la velocidad esperada).
- CPI (Índice de Costo): $EV / AC = 4,000 / 5,500 = 0.73$ (Por cada \$1 invertido, obtenemos \$0.73 de valor).

Pronósticos (Forecasts)

- EAC (Estimado a la conclusión): $BAC / CPI = 10,000 / 0.73$ approx \$13,698 (El proyecto costará más).
- VAC (Variación a la conclusión): $BAC - EAC = 10,000 - 13,698$ approx -\$3,698 (Déficit proyectado).
- TCPI (Índice de desempeño para completar): $(BAC - EV) / (BAC - AC) = (6,000) / (4,500) = \$ 1.33$.
 - Interpretación: El equipo debe trabajar un 33% más eficientemente para recuperar el presupuesto original .

3. Métricas Ágiles de Flujo

A. Velocidad Histórica (Story Points)

- Sprint 1: 15 pts
- Sprint 2: 18 pts
- Sprint 3: 14 pts (Caída por deuda técnica o bugs).
- Promedio: ~15.6 pts.
- Proyección: Si faltan 60 pts, necesitamos approx 4 sprints más .

B. Diagrama de Flujo Acumulativo (CFD)

- Observación: La banda de "In Progress" (En progreso) se está ensanchando.
- Diagnóstico: Cuello de botella. Posiblemente hay muchas tareas de UI iniciadas pero no terminadas por revisiones de PRs lentas o bloqueos de SonarCloud .
- Acción: Limitar el WIP (Work In Progress) a 2 tareas por desarrollador .

C. Throughput (Tasa de entrega)

- Promedio de PRs mergeados a `develop` por semana: 8 PRs.

4. Calidad y DevOps

Métricas de Calidad (Producto)

1. Densidad de Defectos: 0.5 defectos por KLOC (mil líneas de código).
 - Meta: < 0.3. Actualmente estamos un poco altos debido a validaciones de formularios fallidas (RF-01) .
2. Deuda Técnica (SonarCloud): 4 días.
 - Causa: Duplicación de código en los `ViewModel` y reglas de `ktlint` ignoradas inicialmente .

Métricas DORA (Proceso DevOps)




1. Frecuencia de Despliegue: 2 veces por semana (Merge a `develop`).
 - Nivel: Medio (Deseable: Diario).
2. Tasa de Fallos en Cambios (CFR): 15%.
 - Causa: El 15% de los merges rompen el build o fallan los tests unitarios en el pipeline de CI.
 - Meta: < 10% .


5. Dashboards RAG por Stakeholder

Aquí definimos qué ve cada interesado con un código de colores (Rojo/Ámbar/Verde).

Dashboard Ejecutivo (Para el Profesor/Sponsor)




Enfoque: Salud general, presupuesto y tiempos.

Indicador	Valor	Estado (RAG)	Comentario
SPI (Cronograma)	0.80	 Rojo	Retraso acumulado en módulo de Catálogo.
CPI (Costo/Esfuerzo)	0.73	 Rojo	Consumo excesivo de horas en UI Compose.
Riesgos Críticos	2	 Ámbar	R-02 (Curva aprendizaje) materializado.

Alcance Entregado	40%	 Verde	Funcionalidades core (Auth) listas.
-------------------	-----	---	-------------------------------------




Dashboard de Producto (Para PO - Manuel Serna)


Enfoque: Funcionalidad y predicción.

Indicador	Valor	Estado (RAG)	Comentario
Velocidad Prom.	15 pts	 Ámbar	Inestable en últimos sprints.
Sprint Goal	80%	 Verde	Mayoría de historias "Must" entregadas.
Bug Leakage	3 bugs	 Verde	Pocos bugs escapan a producción.

Dashboard Técnico (Para Equipo Dev - Santiago, Adrian, etc.)

Enfoque: Calidad de código y flujo.

Indicador	Valor	Estado (RAG)	Comentario
SonarQube Gate	Failed	 Rojo	Cobertura < 80% en lógica de negocio.
Build Time	3 min	 Verde	Pipeline de GitHub Actions rápido.
WIP	12 items	 Ámbar	Demasiadas tareas abiertas simultáneamente.

CFR (Fallos)	15%	 Ámbar	Mejorar tests locales antes de push.
--------------	-----	---	--------------------------------------

6. Plan de Acciones (Respuesta a Desvíos)

Escenario (Desviación)	Umbral	Acción Correctiva (Responsable)
SPI bajo (Retraso)	< 0.85	Priorización despiadada: Mover historias "Should" (ej. Gestión Trabajadores RF-10) al backlog futuro para asegurar los "Must". (PO).
CPI bajo (Ineficiencia)	< 0.85	Pair Programming: Los expertos en Compose (si hay) deben programar con los que tienen dificultades para reducir tiempos de re-trabajo. (Tech Lead).
Calidad Baja (Sonar)	Gate Fail	Sprint de Hardening: Dedicar el próximo sprint a pagar deuda técnica y subir cobertura, sin añadir features nuevas. (Equipo).
CFR Alto (Builds rotos)	> 10%	Git Hooks: Instalar pre-commit hooks que corran <code>ktlint</code> y <code>testDebugUnitTest</code> localmente antes de permitir el commit. (DevOps).