

UNIVERSIDAD DEL VALLE DE GUATEMALA

Ingeniería de Software – Sección 20

Ing. Pablo Barreno Koch



Documento de reflexión

Osman Emanuel de León García – 23428

Milton Giovanni Polanco Serrano - 23471

Gadiel Amir Ocaña Véliz - 231270

Guatemala, 21 de noviembre de 2025

1. Evolución del Equipo a través del Tiempo

1.1 Línea Temporal del Desarrollo

El proceso de desarrollo de MoneyFlow se caracterizó por tres etapas claramente diferenciadas:

Etapla Inicial (Sprints #1-#4): Marcada por una curva de aprendizaje pronunciada y la búsqueda de roles definidos. La velocidad del equipo fluctuó entre 81.8% y 93.7%, reflejando la adaptación a metodologías ágiles y tecnologías nuevas. Las reuniones de planificación eran extensas (2-3 horas) y la comunicación predominantemente formal.

Etapla de Consolidación (Sprints #5-#7): El equipo alcanzó mayor estabilidad con velocidades entre 82.5% y 96.4%. Se implementaron funcionalidades complejas como notificaciones, backup automático y el módulo de impuestos. Sin embargo, enfrentamos desafíos significativos con la acumulación de deuda técnica y subestimación de tareas.

Etapla de Madurez (Sprints #8-#11): Los últimos cuatro sprints demostraron la madurez alcanzada, sosteniendo 100% de velocidad en los sprints #8, #9, #10 y #11. Se implementaron características avanzadas (IA, CI/CD, seguridad OWASP) con comunicación eficiente (daily stand-ups de 15 minutos) y código de alta calidad (92% test coverage).

1.2 Momentos de Mayor Cohesión

Los momentos de mayor unidad coincidieron con crisis técnicas que requirieron colaboración intensiva:

- **Sprint #11 - Crisis Gemini API:** Cuando se agotó la cuota de Google Gemini (Error 429), el equipo respondió migrando completamente a OpenAI en menos de 11 horas, demostrando capacidad de adaptación y trabajo coordinado.
- **Sprint #10 - Seguridad OWASP:** La detección de vulnerabilidades críticas movilizó al equipo para mejorar el puntaje de seguridad de 42/100 a 78/100 en una semana (+86% de mejora).
- **Sprint #9 - Implementación CI/CD:** La curva de aprendizaje de GitHub Actions fue enfrentada colectivamente, resultando en un pipeline que redujo el tiempo de deployment de 45 minutos a 4.5 minutos (-90%).

Estos momentos compartieron características comunes: objetivos urgentes y claros, impacto visible inmediato, y la necesidad de conocimiento de todos los miembros del equipo.

1.3 Momentos de Mayor Desafío

Los periodos de menor cohesión estuvieron asociados a planificación deficiente y dependencias externas:

- **Sprint #2:** Sobrecarga de trabajo (33 puntos planificados, 27 completados - 81.8%) generó comunicación reactiva y frustración documentada en retrospectivas.

- **Sprint #4:** Bloqueos por Microsoft Auth resultaron en 85.2% de velocidad y reasignación tardía de tareas.
- **Sprint #6:** La complejidad subestimada de notificaciones produjo un PR de 450 líneas que causó conflictos de merge y reviews pendientes por más de 24 horas.

Identificamos un patrón claro: la desunión correlacionó directamente con velocidades bajas (promedio 83.2%), mayor cantidad de bugs (3-5 por sprint), y decisiones retrasadas 2-3 días.

1.4 Impacto en Productividad

El análisis cuantitativo evidencia cómo la cohesión del equipo afectó directamente los resultados:

Nivel de Unidad	Velocidad	Productividad	Bugs en Producción
Alta (Sprints #8-#11)	100%	2.4 pts/h	0
Media (Sprints #3,#5,#7)	92.9%	2.1 pts/h	1-2
Baja (Sprints #2,#4,#6)	83.2%	1.7 pts/h	3-5

Los momentos de alta unidad se caracterizaron por decisiones el mismo día, code reviews en menos de 4 horas, y una moral elevada reflejada en las retrospectivas. En contraste, los periodos de baja cohesión presentaron decisiones retrasadas, trabajo en silos, y soluciones rápidas que generaron deuda técnica.

1.5 Maduración del Equipo

La evolución del equipo es cuantificable a través de múltiples indicadores:

Técnicos:

- Test coverage: 60% → 92% (+53%)
- Technical debt ratio: No medido → 2.8% (excelente)
- Tiempo de deployment: 45 min → 4.5 min (-90%)
- Bugs en producción: 3-5/sprint → 0 (últimos 3 sprints)

De Proceso:

- Planificación: 2-3 horas imprecisas → 1.5 horas precisas
- Daily stand-ups: Irregulares → 15 min consistentes (98% asistencia)
- Code reviews: >24 horas → <4 horas
- Documentación: Al final → Continua (Definition of Done)

Esta maduración se reflejó especialmente en la capacidad de respuesta ante crisis. En el Sprint #9, resolvimos un merge conflict con rollback en 2 minutos sin afectar usuarios, comparado con bloqueos de días en sprints iniciales.

2. Fortalezas y Áreas de Mejora

2.1 Fortalezas Identificadas

Resiliencia Técnica: El equipo demostró capacidad excepcional para enfrentar crisis. Tres pivotes técnicos críticos fueron resueltos exitosamente sin comprometer deadlines: la migración Gemini→OpenAI (11h), rollback ante merge conflict (2min), y compliance OWASP en 7 días.

Ownership Compartido: En los sprints finales, el 100% de las tareas fueron completadas por todos los miembros con distribución equitativa (Osman 38%, Milton 33%, Gadiel 29%), evidenciando responsabilidad colectiva sin dependencias críticas en individuos.

Aprendizaje Acelerado: El equipo dominó tecnologías complejas sin experiencia previa: GitHub Actions (1 semana), caché Redis (implementado exitosamente), y sanitización OWASP (mejora de 86% en puntaje de seguridad).

Enfoque en Calidad: La decisión deliberada de priorizar calidad sobre velocidad en sprints finales resultó en 142 pruebas de regresión voluntarias (100% aprobación), 0 defectos en producción durante 3 sprints, y un SUS Score de 82.5/100 ("Excelente").

2.2 Áreas de Mejora

Estimación de Tareas Nuevas: Subestimamos consistentemente tecnologías sin experiencia previa (CI/CD: +16% sobre estimación, seguridad OWASP requirió buffer no contemplado). Esto generó una desviación presupuestaria final de +8.4%.

Testing en Dispositivos Reales: Durante todo el proyecto trabajamos únicamente con emuladores, generando el riesgo de diferencias no detectadas en producción. Esta limitación fue identificada pero no resuelta.

Documentación Concurrente: Documentar al final de los sprints causó confusión con stakeholders y presión en fechas límite. Solo en el Sprint #11 implementamos documentación como parte del Definition of Done.

Comunicación con Stakeholders: Realizar demos únicamente al final del sprint generó riesgo de feedback tardío. No implementamos ciclos de retroalimentación intermedios hasta las etapas finales.

Gestión de Pull Requests: Un PR de 450 líneas en el Sprint #6 causó conflictos de merge por dificultad de revisión. Establecimos límite de 200 líneas solo después de este incidente.

3. Evaluación de Resultados

3.1 Cumplimiento de Objetivos

Indicador de Éxito del Proyecto:

$(464 \text{ puntos realizados} \times 100) / 437 \text{ puntos planificados} = 106.2\%$

Este resultado indica una sobre-entrega del 6.2%, cumpliendo todos los objetivos planificados y agregando 27 puntos de funcionalidades adicionales de alto valor (CI/CD y Seguridad OWASP).

Análisis de Completitud:

- 27 de 29 historias de usuario implementadas (93%)
- 2 historias en progreso (deuda técnica y refinamientos)
- 0 historias eliminadas del backlog original
- 8 historias modificadas por mayor complejidad detectada
- 2 historias completamente nuevas agregadas

Métricas de Calidad Alcanzadas:

- System Usability Scale: 82.5/100 (objetivo: >70)
- Test Coverage: 92% (objetivo: >80%)
- Performance: 400 usuarios concurrentes (objetivo: 200)
- Seguridad: 0 vulnerabilidades críticas
- Uptime: 96% (27/28 deploys exitosos)

3.2 Suficiencia de Puntos Realizados

Consideramos que los 464 puntos realizados son suficientes para un **MVP académico completo**, pero requieren 25-30 puntos adicionales para producción real:

Funcionalidad Core: 100% Completa

- Sistema de autenticación y usuarios
- Gestión completa de transacciones (ingresos/gastos)
- Reportes con visualizaciones y exportación PDF
- Sistema de consejos financieros con IA
- Homepage profesional y responsive

Pendientes Críticos para Producción:

- MFA (Autenticación Multifactor) - 8 puntos
- Monitoreo con Sentry - 4 puntos
- Políticas de privacidad y términos legales - 5 puntos
- Penetration testing externo - 8 puntos

El sistema actual es completamente funcional y de calidad profesional, pero necesita 4-6 semanas adicionales para cumplir estándares de producción comercial.

3.3 Planificación vs. Ejecución

Diferencias Principales:

Aspecto	Planificado	Ejecutado	Variación
Duración	~8 meses	~11 meses	+37%
Puntos de Historia	437	464	+6.2%
Presupuesto	Q31,100	Q33,700	+8.4%
Funcionalidades	16 core	29 total	+81%

Funcionalidades No Planificadas Inicialmente:

1. **Sistema de IA** (27 puntos): No estaba en el scope inicial pero se convirtió en el diferenciador principal del producto.
2. **CI/CD Enterprise-Grade** (12 puntos): Planificamos deployment simple, ejecutamos un pipeline profesional que ahorra 95% del tiempo.
3. **Seguridad OWASP** (20 puntos): No contemplada inicialmente, resultó crítica para viabilidad en producción.
4. **Módulo de Impuestos** (15 puntos): Agregado para localización al mercado guatemalteco.

La planificación original fue optimista en cuanto a tiempos pero conservadora en alcance. El incremento del 81% en funcionalidades justifica la extensión del 37% en duración, manteniendo una desviación presupuestaria aceptable del 8.4%.

4. Conclusión: Éxito del Proyecto

4.1 Valoración Final

Consideramos que MoneyFlow fue un **proyecto exitoso** basándonos en tres dimensiones:

Cuantitativamente: El 106.2% de cumplimiento, combinado con 100% de velocidad sostenida durante los últimos tres sprints y 0 bugs críticos en producción, demuestra excelencia en ejecución.

Cualitativamente: Entregamos un sistema funcional end-to-end con calidad profesional (no prototipo académico), infraestructura escalable con CI/CD automatizado, y documentación exhaustiva que permitiría transferencia a otro equipo.

Comparativamente: Superamos expectativas en performance (200% del objetivo), innovación (IA no planificada), calidad de UX (SUS 82.5 vs 70 objetivo), e infraestructura (enterprise-grade vs básica planificada).

4.2 Aprendizajes Clave

Lo que funcionó:

- Metodología ágil con retrospectivas accionables
- Pair programming ante desafíos técnicos complejos
- Priorización de calidad sobre velocidad en etapas finales
- Comunicación multi-canal (WhatsApp, Zoom, presencial)

Lo que mejoraríamos:

- Incluir infraestructura (CI/CD) desde Sprint #1
- Buffer del 30% para tecnologías nuevas
- Testing UX desde etapas tempranas
- Documentación como Definition of Done desde inicio
- Demos semanales con stakeholders

4.3 Reflexión del Equipo

Después de casi un año de desarrollo, el equipo MoneyFlow evolucionó de tres estudiantes aprendiendo metodologías ágiles a un equipo maduro capaz de implementar características enterprise-grade. La velocidad del 100% en los últimos sprints no refleja solo eficiencia técnica, sino la confianza mutua y los procesos consolidados que desarrollamos.

Los momentos de mayor desafío —la crisis de Gemini API, las vulnerabilidades OWASP, los conflictos de merge— no nos fracturaron sino que fortalecieron nuestra capacidad de respuesta colectiva. Aprendimos que la verdadera madurez de un equipo no se mide en la ausencia de problemas, sino en la velocidad y calidad de su resolución.

MoneyFlow representa más que un proyecto académico exitoso; es evidencia tangible de que tres estudiantes pueden, con la metodología correcta y el compromiso adecuado, crear un sistema de calidad profesional listo para impactar usuarios reales.