Codelab:

Proceso con enfoque ADD y Clean Architecture

Desarrollo de Software III

Henao Aricapa, STIVEN – 2259603

Docente:

Salazar, ÁLVARO

Sede Tuluá

Junio de 2025

1. ¿Qué es Attribute-Driven Design (ADD) y cuál es su propósito en el diseño de software?

ADD (Attribute-Driven Design) es una metodología de diseño arquitectónico centrada en los *atributos de calidad* del sistema, como rendimiento, seguridad, escalabilidad, disponibilidad, entre otros. Su propósito es tomar decisiones de diseño arquitectónico fundamentadas en las prioridades no funcionales del sistema, garantizando que la arquitectura cumpla con los requisitos críticos de calidad.

2. ¿Cómo se relaciona ADD con Clean Architecture en el proceso de diseño de sistemas?

ADD y Clean Architecture se complementan en distintas fases del diseño. ADD se aplica al inicio para definir *qué* necesita cumplir el sistema en términos de calidad, mientras que Clean Architecture se encarga de *cómo* se implementa esa arquitectura, asegurando separación de responsabilidades y un sistema desacoplado que facilite el mantenimiento y la escalabilidad.

3. ¿Cuáles son los pasos principales del método ADD para definir una arquitectura de software?

Los pasos clave en ADD son:

1. Identificar requisitos y atributos de calidad prioritarios.
2. Establecer restricciones tecnológicas.
3. Seleccionar tácticas arquitectónicas para cada atributo de calidad.
4. Definir los módulos principales y sus interacciones.
5. Validar la arquitectura mediante revisiones y pruebas.

4. ¿Cómo se identifican los atributos de calidad en ADD y por qué son importantes?

Los atributos de calidad se identifican a partir de los requisitos del sistema y las expectativas del negocio o cliente. Se priorizan según su impacto en la experiencia del usuario y el éxito del sistema. Son importantes porque orientan las decisiones de diseño hacia cumplir objetivos como tiempo de respuesta, disponibilidad del servicio, seguridad de la información, etc.

5. ¿Por qué Clean Architecture complementa ADD en la implementación de una solución?

Clean Architecture permite traducir las decisiones tomadas con ADD en una estructura de código bien organizada. Al dividir el sistema en capas (Dominio, Aplicación, Infraestructura, Presentación), facilita la implementación de tácticas de calidad como desacoplamiento, mantenibilidad y escalabilidad, respetando los principios definidos previamente con ADD.

6. ¿Qué criterios se deben considerar al definir las capas en Clean Architecture dentro de un proceso ADD?

- Separación de responsabilidades.
- Independencia de tecnologías y frameworks.
- Inversión de dependencias (el dominio no depende de detalles externos).
- Reutilización y testabilidad del código.
- Soporte para atributos de calidad como rendimiento (mediante capas optimizadas) o seguridad (control de acceso en la capa adecuada).

7. ¿Cómo ADD ayuda a tomar decisiones arquitectónicas basadas en necesidades del negocio?

ADD traduce las necesidades del negocio en atributos de calidad medibles. A partir de estos, el arquitecto selecciona tácticas adecuadas (ej. usar caché para rendimiento, encriptación para seguridad), asegurando que cada decisión técnica tenga un respaldo en los objetivos estratégicos del sistema.

8. ¿Cuáles son los beneficios de combinar ADD con Clean Architecture en un sistema basado en microservicios?

- Diseño enfocado en calidad desde el inicio.
- Servicios desacoplados, orientados a casos de uso.
- Alta mantenibilidad y facilidad de prueba.
- Capacidad de adaptar o escalar servicios sin romper la lógica del dominio.
- Facilidad para validar que cada microservicio cumple sus atributos clave (como disponibilidad o rendimiento).

9. ¿Cómo se asegura que la arquitectura resultante cumpla con los atributos de calidad definidos en ADD?

Se realiza una fase de **validación y refinamiento**, que incluye:

- Revisiones arquitectónicas.
- Pruebas de rendimiento, carga y seguridad.
- Análisis de cuellos de botella.
- Ajustes de implementación o diseño (ej. agregar caching, mejorar consultas, replicar servicios).

10. ¿Qué herramientas o metodologías pueden ayudar a validar una arquitectura diseñada con ADD y Clean Architecture?

- Revisiones por pares o arquitectos (ARB - Architecture Review Board).
- Pruebas de carga y estrés (ej. JMeter, Gatling).
- Pruebas de seguridad (ej. OWASP ZAP, SonarQube).
- Análisis estático de código (ej. SonarQube, PMD).
- Monitoreo y observabilidad (ej. Prometheus + Grafana).
- Modelos de evaluación de arquitectura como ATAM (Architecture Tradeoff Analysis Method).