# Introducción

Los modelos de procesos de desarrollo de software se pueden dividir en dos categorías los modelos tradicionales y los modelos agiles:

Los modelos tradicionales son Cascada y sus extensiones, prototipado, desarrollo rápido de aplicaciones (RAD), espiral, proceso unificado (UP) e incremental. Los modelos agiles son programación extrema (XP), scrum, métodos de desarrollo de sistemas dinámicos (DSDM), la Desarrollo guía por funcionalidades (FDD), el desarrollo de software adaptativo (ASD) y cristal.( Adopting Logical Architectures within Agile Projects)

Los modelos tradicionales se enfocan en tener una documentación pesada a través de las fases del desarrollo de software, introduciendo retrasos en el desarrollo de software. Por otra parte, estos modelos no admiten la comunicación frecuente entre los diferentes stakeholders y los desarrolladores de software luego de que los requerimientos fueron capturados, debido a que no se promueve las modificaciones en los resultados finales. ( Adopting Logical Architectures within Agile Projects)

Del otro lado, los modelos ágiles son procesos ligeros se concentran en menos documentación para reducir los retrasos, que fue un inconveniente importante de los modelos tradicionales. Otra característica de los modelos ágiles es que comunicación es fuerte y frecuente entre las stakeholders y los desarrolladores de software durante todo el proceso de desarrollo de software. Debido a esta característica, se pueden hacer modificaciones en los requisitos durante el desarrollo. Los modelos ágiles proporcionan entorno de desarrollo rápido mediante la superposición de algunas de las fases tradicionales de desarrollo de software [1]. ( Adopting Logical Architectures within Agile Projects)

La arquitectura se define comúnmente como “la descomposición estructural de un sistema, incluida su descomposición en partes, su conectividad, los mecanismos de interacción y los principios rectores que informan el diseño del sistema”.

Aunque no es incorrecta desde una perspectiva técnica, esta definición está abierta a un amplio espectro de interpretación. Puede significar cualquier cosa, desde un boceto de diseño de alto nivel con poca relación con la tecnología, el código o el sistema real que se está construyendo hasta un diseño grande y rígido con muchos detalles de clase y nivel de código. En la práctica, ninguna de las vistas ofrece una guía suficiente para el proceso real de desarrollo, una contribución necesaria a la arquitectura "buena". La primera forma es demasiado abstracta para dar una orientación concreta; El segundo es demasiado prematuro, decisivo antes de que se conozcan los detalles relevantes. Por lo tanto, no es sorprendente que algunos en la comunidad ágil sostengan que la arquitectura no es una preocupación central en el desarrollo de software práctico. (Architecture and Agility: Married, Divorced, or Just Good Friends?)

# Desarrollo del trabajo

## Métodos de evaluación de arquitectura de software

Los profesionales que usan métodos ágiles ven los métodos centrados en la arquitectura como un esfuerzo de documentación pesada (No van de acuerdo a los esfuerzos del proyecto, si durante el proyecto cambian representa más esfuerzo durante el desarrollo) y que la arquitectura debe emerger gradualmente al ejecutar el sprint, como resultado de la refactorización sucesiva [3]. por otro lado, [4] presenta un estudio donde los desarrolladores ágiles perciben la arquitectura de software como un aspecto relevante, tales como la comunicación entre los miembros del equipo, para fundamentar las decisiones de diseño, la documentación de las hipótesis de diseño y evaluación de alternativas. (Adopting Logical Architectures within Agile Projects)

Al igual que en cualquier tipo de proyectos de software, la buena arquitectura de software es muy importante ya que es el marco fundamental de todas las decisiones técnicas. En un contexto ágil, argumentan que la arquitectura se utiliza para proporcionar los requisitos, tanto funcionales como no funcionales. (Adopting Logical Architectures within Agile Projects)

El enfoque de Madison llamada arquitectura ágil, aboga por la coexistencia de ágil y arquitectura como enfoques complementarios aplicando adecuadamente las combinaciones adecuadas de las funciones de arquitectura (como la comunicación, los atributos de calidad, y los patrones de diseño) y conocimientos de arquitectura en el ciclo de vida de desarrollo. La literatura muestra otros trabajos de investigación, es decir, que presentan propuestas marco que exploran la relación y las sinergias entre los métodos de análisis y diseño centrado en la arquitectura del Software Engineering Institute (SEI) que el diseño de la arquitectura respecto y revisión dentro de los marcos ágiles. Jeon et al. proponen un atributo personalizado Taller de Calidad (QAW) y Diseño guiado por atributos (ADD) se utiliza en proyectos Scrum. (Adopting Logical Architectures within Agile Projects)

Existen muchos métodos para la evaluación de la arquitectura de software y la implementación de estos métodos en desarrollos agiles es posible: un SAEM híbrido para FDD de modo que las actividades de evaluación de la arquitectura se pueden realizar y sin embargo, la agilidad no se vea comprometida. Un modelo híbrido se compone de tres SAEMs es decir; QAW, ATAM y ARID. Y este enfoque también se puede extender a otros modelos de procesos ágiles. ( Adopting Logical Architectures within Agile Projects)

Arquitectura Agil

Para definir una buena arquitectura, cuántas cantidades diferentes de eventualidades deben tomarse en consideración y, por lo tanto, cuánto tiempo y esfuerzo se debe invertir en el diseño e implementación de componentes con respecto a un buen diseño arquitectónico para cubrir todas las posibilidades, que eventualmente puede no ser usado en absoluto. Por otro lado, no hay planificación previa o casi ninguna, y, al mismo tiempo, se corre el riesgo de rediseñar el diseño arquitectónico existente desde cero, cada vez que no es capaz de manejar el último requisito. En definitiva, significa que la arquitectura y los negocios no evolucionan de la misma manera y a la misma "velocidad". Los problemas relacionados con los problemas de arquitectura y diseño en desarrollo de software ágil se han discutido en varios artículos y afirman que ASD no presta atención a los problemas de arquitectura y diseño y, por lo tanto, está obligado a generar Decisiones de diseño subóptimas. (Towards an Architectural Framework for Agile Software Development)

## “Arquitectura simplemente no se puede hacer de forma incremental.”

Suponiendo que el nivel de granularidad mencionada no se considera trivial, podemos afirmar esta declaración casi siempre puede ser refutada. El error de un equipo de proyecto son que los cambios tienen que ser terminados completamente y listo para su consumo masivo en el lanzamiento. Un enfoque en la forma en que sus clientes realmente funcionan puede ayudar a proporcionar la orientación sobre lo que realmente se necesita. Dado una idea de cómo se desea que sea el diseño final, no siempre es necesario que todos los componentes sean refactorizados al mismo tiempo. Dividir y conquistar en iteraciones los componentes que forman parte del diseño, utilizando andamiaje temporal como (clases proxy, adaptadores, etc.) adecuados para vincular los componentes antiguos a la nueva arquitectura.

(Agile Architecture IS Possible – You First Have to Believe! )

Qué significa la arquitectura para el desarrollo ágil, en contraste con algunas discusiones que observamos en la comunidad de software, el desarrollo ágil no se trata de la adopción por parte del culto a Scrum o cualquier otro proceso, conjunto de herramientas o metodología, aunque ciertamente observamos esto y lo consideramos un problema. La esencia de la agilidad es la capacidad de respuesta, el aprendizaje y la suficiencia. La agilidad se refleja en la sostenibilidad y la calidad del software y su desarrollo; por definición, el desarrollo insostenible y de mala calidad contradice y reduce la agilidad. El Manifiesto para el desarrollo de software ágil señala que "la atención continua a la excelencia técnica y al buen diseño mejora la agilidad", ofreciendo a la arquitectura un papel claro dentro de ágil, aunque no se expresa a través del gran diseño inicial. (Architecture and Agility: Married, Divorced, or Just Good Friends?)

No se que poner

Engineering Values: From Architecture Games to Agile Requirements

Valores ágiles y principios centrados en la arquitectura

Para caracterizar las relaciones entre los valores ágiles y los principios centrados en la arquitectura. Se destilaron tres principios fundamentales de los métodos centrados en la arquitectura:

* Impulsado por requisitos no funcionales,
* Exigir una inversión inicial, y
* Forzando el cumplimiento de la arquitectura del software.

(Peaceful Coexistence: Agile Developer Perspectives on Software Architecture)