

# GenAI for Software architecture

Miguel Olamendi Alonso – UO285032

Saúl Valdelvira Iglesias – UO283685

## 1. Introducción

La inteligencia artificial generativa está transformando la manera en que se desarrolla software, pero su aplicación en arquitectura de software plantea tanto oportunidades como limitaciones. En esta presentación, exploraremos cómo estas herramientas pueden asistir en tareas dentro de la arquitectura de software, basándonos en la entrevista a Ipek Ozkaya, investigadora del SEI en Carnegie Mellon. Ozkaya tiene una amplia trayectoria en la evolución de sistemas y la gestión de la deuda técnica, lo que le permite ofrecer una visión fundamentada sobre el potencial y las limitaciones de la IA en este ámbito.

## 2. ¿Quién es Ipek Ozkaya?

Ipek Ozkaya es investigadora principal en el Software Engineering Institute de Carnegie Mellon University. Su especialización está en la evolución de software a gran escala, con un enfoque particular en la gestión de la deuda técnica y la arquitectura de software. Ha trabajado en metodologías que buscan mejorar la calidad y mantenibilidad de sistemas críticos. A través de su experiencia, ha estudiado cómo las decisiones arquitectónicas impactan en la sostenibilidad de un software a largo plazo y cómo la automatización puede contribuir a estas decisiones.

## 3. Rol de la IA Generativa en Arquitectura de Software

La IA generativa ha demostrado su utilidad en múltiples aspectos del desarrollo de software, pero su impacto en la arquitectura del software es un tema aún en exploración. Una de las principales diferencias radica en que la implementación de código puede automatizarse en cierta medida, mientras que las decisiones arquitectónicas requieren un nivel de abstracción y análisis contextual que la IA aún no puede replicar de manera confiable. En este sentido, la IA puede ser vista como un asistente que ayuda en la ejecución de tareas específicas, pero no como un reemplazo de la toma de decisiones arquitectónicas estratégicas.

## 4. Aplicaciones de la IA Generativa en Arquitectura de Software

Durante la entrevista, Ozkaya menciona que la IA generativa es útil en tareas específicas que implican generación o transformación de código, pero no necesariamente en la toma de decisiones arquitectónicas complejas. Una de las aplicaciones más destacadas es la **documentación automática**, donde la IA puede analizar grandes volúmenes de código y generar explicaciones estructuradas sobre su funcionamiento. También puede ser útil en la **generación de pruebas**, automatizando la creación de casos de prueba a partir del código existente y detectando posibles fallos antes de su implementación. Otra área en la que la IA tiene potencial es en la **refactorización del código**, ayudando a optimizar estructuras de código

sin alterar su funcionalidad, lo que puede ser beneficioso en la reducción de deuda técnica. Además, la IA puede asistir en la **conversión entre lenguajes de programación**, permitiendo la migración de código entre tecnologías sin necesidad de una reescritura manual completa. Sin embargo, todas estas aplicaciones dependen de la validación humana para asegurar que los resultados sean correctos y útiles.

## 5. Limitaciones y Desafíos

A pesar de su potencial, Ozkaya señala varios desafíos que impiden que la IA generativa se convierta en una herramienta totalmente autónoma en la arquitectura de software. En primer lugar, estas herramientas **se centran más en la implementación que en la arquitectura en sí**, lo que significa que pueden generar código, pero no necesariamente respetarán principios arquitectónicos adecuados. Además, hay una **falta de consistencia en las respuestas generadas**, lo que puede hacer difícil confiar plenamente en la IA para la toma de decisiones críticas. Otro problema relevante es la **trazabilidad**, ya que muchas veces la IA no ofrece explicaciones claras de por qué toma ciertas decisiones, lo que complica la validación y el mantenimiento del software a largo plazo. Finalmente, la IA sigue dependiendo en gran medida de la **intervención y supervisión humana**, ya que los resultados pueden ser erróneos, ineficientes o incluso generar más problemas de los que resuelven si no se analizan adecuadamente.

## 6. Impacto en la Toma de Decisiones Arquitectónicas

Uno de los aspectos más debatidos en la entrevista es si la IA puede asistir en la toma de decisiones arquitectónicas o si su papel se limita a la asistencia en tareas más técnicas. Ozkaya menciona que, aunque la IA puede analizar patrones en código y sugerir soluciones, las decisiones arquitectónicas involucran múltiples factores que van más allá del análisis de código. Estos incluyen requerimientos de negocio, consideraciones de escalabilidad y la alineación con objetivos estratégicos de la organización. La IA, en su estado actual, **carece de la capacidad de razonar sobre estos aspectos de manera autónoma**. Sin embargo, su uso como herramienta de apoyo podría evolucionar, especialmente si se desarrollan modelos más avanzados que integren conocimiento arquitectónico y permitan una interacción más explicativa con los desarrolladores.

## 7. La Brecha entre Diseño y Realidad

El diseño arquitectónico se expresa comúnmente mediante diagramas (por ejemplo, UML), pero su implementación final requiere la conversión a código. Esta brecha surge cuando se pierde la trazabilidad entre los diseños de alto nivel y su expresión en el código, lo que puede convertirse en un problema si el código evoluciona de manera independiente al diseño (y viceversa). Actualmente, no existen herramientas que permitan manejar ambos aspectos de manera integrada.

## 8. Niveles de Abstracción

Un arquitecto de software debe comprender el sistema desde diferentes niveles de abstracción. Si bien la IA puede generar funciones e implementar algoritmos, aún se cuestiona su capacidad para entender la estructura del sistema en su totalidad y razonar sobre ella de manera efectiva.

## 9. Los Diseños Arquitectónicos vs. la Realidad

En la práctica, muchas veces es necesario "saltarse las normas" para optimizar el rendimiento. Surge entonces la pregunta: ¿cómo podría la IA discernir cuándo seguir los principios arquitectónicos y cuándo aplicar soluciones no convencionales?

## 10. Calidad de los Resultados de la IA

La calidad de los resultados generados por la IA depende de dos factores principales:

- La precisión de los *prompts* (cómo se formula la pregunta).
- La información con la que se ha entrenado el modelo.

## 11. Limitaciones de los Modelos de Lenguaje (LLM)

Los LLM son modelos probabilísticos, lo que plantea dudas sobre su capacidad para:

- Trabajar con formalismos rigurosos.
- Comprender los aspectos humanos en la arquitectura de software.  
Como señala Ozkaya: "[...] a large-scale system needs to be represented with different levels of abstraction without forgetting the implementation constructs."

## 12. Retos de la IA Generativa

Entre los principales desafíos para su adopción en arquitectura de software destacan:

- El tamaño limitado del contexto que pueden manejar.
- Su naturaleza probabilística, que introduce incertidumbre en los resultados.

## 13. Beneficios Potenciales

A pesar de las limitaciones, la IA generativa ofrece ventajas significativas:

- Puede automatizar tareas repetitivas de desarrollo, liberando tiempo para actividades arquitectónicas estratégicas.
- Su rápida evolución sugiere que su utilidad en este campo aumentará en el futuro.