In 2005, the company I ran was already using utility like infrastructure. We had evolved early DevOps practices — distributed systems, continuous deployment, design for failure — and this was just the norm for us. However, we had also produced the utility coding platform known as Zimki, which happened to allow developers to write entire applications, front and back end in a single language — JavaScript. As a developer you just wrote code, you were abstracted away from the platform itself, you certainly had no concept of servers. That every function you wrote within your program could be running in a different platform stack was something you didn't need to know. From a developer point of view you just wrote and ran your program and it called other functions. However, this environment enabled some remarkable new capabilities from distribution of functions to billing by function. The change of platform from product to utility created new characteristics that enabled new architectural practices to emerge at this level. This is co-evolution. This is normal. These new practices, I've nicknamed FinDev for the time. The "old" best architectural practices, well, that's legacy. I've drawn a map to show this change.

Figure 101 — Co-Evolution of Architectural Practice (platform)