Don’t Elevate the Means Beyond the End

Seth Dobbs

Our industry is rife with trendy technologies such as microservices, serverless, and blockchain, and trendy processes such as Agile and Lean in all of their various forms. Years ago, the trends were EJBs and UML, and before that, shifting to object orientation and having a structured Waterfall were important.

In their time, none of these were bad in and of themselves, but each wave creates the potential for missing the point behind each of these waves and leads to organizations serving the means. This behavior manifests in comments such as “that’s a bad requirement because it doesn’t fit our architecture” and “that’s not Agile!” In fact, we as an industry can become dogmatic around the means as if *that* is our purpose rather than them merely being tools.

For example, I’ve encountered several organizations with the directive to “implement microservices.” The problem is, “not having microservices” isn’t a problem, *per se*, nor is microservices a solution in and of itself. It is a tool or a means for solving a problem. This becomes more ironic in organizations that are dogmatic about Agile given that the *Agile Manifesto* is itself a set of principles that among other things eschews dogmatic process.

Not to pick on microservices specifically; this is an architectural approach that provides a ton of value for certain problem spaces, just as EJBs, RPCs, remote SQL, and various other technologies and techniques have in the past. Which is precisely the point; none of these were the final approach to building software, yet each time we as an industry are faced with a new approach, it becomes *the* solution, even though many of us have been around long enough to know things will change again.

Put simply, what does it mean to replatform to microservices? How do we know whether we've done it correctly? In the absence of an actual business problem we're trying to solve, it's hard to actually measure the value of what we've done. Which is why we as technology leaders should embrace the following principle:

> Don't elevate the means beyond the end.

In other words, we need to remember that we are solving problems for our business and for our users and that, ultimately, our approaches have value only when they achieve an end.

As an industry, we often have a tendency to elevate the means; learning new technologies is a big reason why many of us are in this field, after all. And sometimes our development teams feel distant from the overall value of the company we work for, so we focus on what we can control to make our lives interesting.

However, I believe that feeling that our development work is well-aligned with the overall success of the companies we work for is a very rewarding feeling and provides the greatest motivation for our teams.

Putting this into practice means understanding that our business needs don't serve our architecture but rather that our architectures enable us to achieve something great.

Each of the approaches mentioned at the start are (or were) good at solving specific problems and were considered the modern approach in their time, but modernizing an architecture in and of itself is not an end. "Modernizing" doesn't give us guidance into how to best leverage microservices or whatever the next wave will be. Understanding the end goals—be it organizational agility, horizontal scalability to minimize downtime at high loads, or some other need, will help us make better decisions on how to approach our work.

If we can understand when we should and shouldn't use new technologies, how they better solve problems than past approaches, and how to apply them to solve problems facing our companies, we will be truly successful.