Expert Insight Video: 3 Things You Need to Know About Microservices

Published 5 July 2021 - ID G00753715 - 1 min read

By Analyst(s): Anne Thomas

Initiatives: Software Engineering Strategies

Hype is rampant around microservices, often including significant misinformation and misunderstanding. This video cuts through the hype and discusses the agility and complexity trade-off that comes with microservices, enabling software engineering leaders to make informed decisions about their use.

Overview

Microservices architecture offers powerful benefits, including agility, scalability and resilience from both a technical and business perspective. But the price for these benefits is high. Microservices architecture is complex and culturally disruptive. And in most circumstances, microservices architecture is overkill for an application. This video describes three things that software engineering leaders need to know about microservices in order to make informed decisions:

1. Microservices are tools that enable better agility and scalability. Don't think of them as the ultimate destination. You need to weigh the trade-off between agility and complexity. Only use microservices when they deliver enough value to justify the increase in complexity.

2. Microservices won't save you money. In fact, reaping the benefits of a microservices architecture will require significant investment.

3. Microservices are culturally disruptive. To use them successfully, you must adopt a product mindset. Your teams must use agile and DevOps methods and continuous delivery practices. You also need to change your team structures to align with product boundaries. If you can't make these cultural changes, you should not use microservices.
Recommended by the Authors

Leading Teams to Success With Microservices Architecture

Designing Services and Microservices to Maximize Agility

10 Ways Your Microservices Adoption Will Fail – And How To Avoid Them

Working With Data in Distributed and Microservices Architecture

Quick Answer: What Essential Patterns Should Microservices Follow for Successful Use in Production?