Complexity is killing software developers

The growing complexity of modern software systems is slowly killing software developers. How can you regain control, without losing out on the best these technologies have to offer?

By Scott Carey
UK Group Editor, InfoWorld
NOV 1, 2021 3:00 AM PDT

“Complexity kills,” Lotus Notes creator and Microsoft veteran Ray Ozzie famously wrote in a 2005 internal memo. “It sucks the life out of developers; it makes products difficult to plan, build, and test; it introduces security challenges; and it causes user and administrator frustration.”

If Ozzie thought things were complicated back then, you can’t help but wonder what he would make of the complexity software developers face in the cloud-native era.

[Also on InfoWorld: Why you should use a microservice architecture]

The shift from building applications in a monolithic architecture hosted on a server you could go and touch, to breaking them down into multiple microservices, packaged up into containers, orchestrated with Kubernetes, and hosted in a distributed cloud environment, marks a clear jump in the level of complexity of our software. Add to that expectations of feature-rich, consumer-grade experiences, which are secure and resilient by design, and never has more been asked of developers.

“There is a clear increase in complexity when you move to such a pervasive microservices environment,” said Amazon CTO Werner Vogels during the AWS Summit in 2019. “Was it easier in the days when everything was in a monolith? Yes, for some parts definitely.”

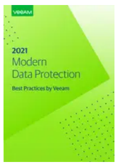
Or, as his colleague, head of devops product marketing at AWS, Emily Freeman, said in 2021, modern software development is “a study in entropy, and it is not getting any more simple.”

On the other hand, complex technologies have never been easier to consume off the shelf, often through a single API—from basic libraries and frameworks, to image recognition capabilities or even whole payments stacks. Simply assemble and build your business logic on top. But is it really that simple?

“It has never been more difficult to be a software developer than it is today,” said Nigel Simpson, a consultant and former director of enterprise technology strategy at Walt Disney. “While we’ve seen an up-leveling of capabilities that enable developers to do more by using high-level frameworks for application development and machine learning, this comes at a cost. The explosion of choice and the pace of development make it challenging for developers to keep up with the zeitgeist, with many developers getting caught in the headlights.”

RECOMMENDED WHITEPAPERS

KnowBe4 Human error. Conquered. Critical Considerations When Choosing Your Security Awareness Training Vendor


VEEAM
2021 Modern Data Protection
Best Practices by Veeam

2021 Modern Data Protection Best Practices by Veeam

10 WAYS A ZERO TRUST ARCHITECTURE PROTECTS AGAINST RANSOMWARE

10 Ways a Zero Trust Architecture Protects Against Ransomware

Essential vs. accidental complexity

Justin Etheredge, cofounder of the software agency Simple Thread, helpfully differentiates between essential and accidental complexity. He told InfoWorld, “Essential is the complexity in the business domain you are working in, the fact that enterprises are extremely complicated environments, so the problems they are trying to solve are inherently complex. The other area is accidental; this is the complexity that comes with our tooling and what we layer on top when solving a problem.”

The cloud-native era has ushered in the potential for more accidental complexity than ever before, setting a collision course between developers, who want to leverage the full toolkit available to them, and their bosses, who want them to focus on delivering value to customers.

"Given the demand for software developers today, companies don't have the leverage to push developers towards a mental model of primarily delivering value to their customers," Etheridge said. "Getting more engineers to think that way is a challenge."

The downside of choice

The combined popularity of cloud computing and the open source software movement has seen the number of options available for developers to build and run more scalable, resilient, modular, and updatable applications rise at an inexorable rate.

"Everything was so much simpler before, not because we made a mistake as an industry, but because the demand of those systems grew so dramatically that we have to ship faster," said Kaspar von Grünberg, founder of Humanitec, a startup that helps companies build their own developer platforms, in an interview with InfoWorld.

The Cloud Native Computing Foundation (CNCF) maintains an interactive graphic of the nearly 1,000 unique services that make up the cloud-native ecosystem, many of which are free and open source to boot. Additionally, each of the big three cloud providers—Amazon Web Services, Microsoft Azure, and Google Cloud—offers about 200 unique services to customers, across compute, storage, database, analytics, networking, mobile, developer tools, management tools, IoT, security, and enterprise applications.

"The process of application development is simply too fragmented at this point; the days of every enterprise architecture being three-tier, every database being relational, and every business application being written in Java and deployed to an application server are over," wrote RedMonk analyst Stephen O'Grady in a 2020 blog post. "The single most defining characteristic of today's infrastructure is that there is no single defining characteristic. It's diverse to a fault."

Or, as the ex-Tumblr CTO Marco Arment wrote back in 2015, “Web development has never been more complicated or convoluted than it is today due to the sheer quantity of tools (and their rapid rate of change) involved in most modern web-dev environments.”

One of the consequences of the tried-and-tested approach cloud vendors have taken to product development—small, independent, two-pizza teams building services in response to customer demands—is that developers have been largely empowered to choose how to assemble this multitude of building blocks together in a way that delivers business value.

[Also on InfoWorld: No one wants to manage Kubernetes anymore]

“You are like a kid in the candy shop in the cloud,” said Camille Fournier, head of platform engineering at financial services firm Two Sigma, in an interview with InfoWorld. “But as you grow and try to make things fit together, the complexity absolutely multiplies.”

Which leads many to question whether this level of choice is a net positive for the average software developer. Or, as O’Grady concluded in that 2020 blog post, “The complexity inherent to a huge catalog of available services can become, in certain settings, less a strength than a liability.”

Let’s build an internal platform

This growing level of complexity has led many organizations to adopt a central platform model, where an internal platform team is tasked with vetting the tools most required by engineers, building templates, and plotting golden paths to ease their journey into production, while also centralizing functions like financial operations, security, and governance to ease the cognitive load on individual developers.

Take the music streaming giant Spotify as an example. “Rolling back six or so years, Spotify was (and still is) committed to an agile engineering culture with autonomous teams,” Spotify product manager Gary Niemen wrote in a 2020 blog post. “With all the advantages that brings, it also brought forth complexities, including a fragmented ecosystem of developer tooling where the only way to find out how to do something was to ask your colleague."

As Spotify scaled, it found that the approach that had powered its rapid growth was actually starting to slow it down. It needed to consolidate and simplify. "The blessed or recommended tooling should be easily discoverable. The journey through that tooling should be clear. There should be quality user instructions along the way. And, if users get stuck, where to get support should be obvious," Niemen wrote.

The key to a good internal developer platform then is finding that balance between self-service for developers who want to get on with the job at hand and abstracting the tasks that are the least valuable, without making developers feel restricted, wrote Humanitec's von Grünberg in a 2021 blog post.

"The idea behind having golden paths is not to limit or stifle engineers, or set standards for the sake of it. With golden paths in place, teams don't have to reinvent the wheel, have fewer decisions to make, and can use their productivity and creativity for higher objectives. They can get back to moving fast," Spotify product manager Niemen wrote.

The problem is, "developers love reinventing the wheel. Nothing satisfies me more than building a better mousetrap," consultant Simpson said. But in a world where a lot of the answers are sitting right there on Stack Overflow, is this the best use of developers' time?

Amanda Silver, CVP of Product, Developer Division, at Microsoft, said, "There will always be some organizations that try to clamp down and others that try to empower developers. The core is the notion of developer velocity. We can build systems where the developer can write the code that only they can write and aren't distracted or burdened to learn domains that aren't differentiated for them."

Founded in 1987, travel technology company Amadeus has lived through these waves of technology change, having built its original applications on the mainframe, shifting to building on top of an open Linux platform in the early 2000s, and now is leaning heavily toward containerized applications orchestrated with Kubernetes.

“Our developers need to be able to develop on top of the core we offer, so the idea is a platform approach where we provide capabilities for them,” Edouard Hubin, head of infrastructure and cloud at Amadeus, told InfoWorld. “New technologies bring more complexity for security and stability. When you open up a system like this you want stability. The rise of data-driven applications is a completely different level of complexity for us…. It brings a new way to write applications and build feedback loops. All these things are new and bring complexity.”

As a result, Hubin wants to hide complexity where he can, either through an internal team designing solutions or by paying for managed services where they make sense. Take databases as an example. Amadeus used to manage its own MongoDB instances, but now uses the vendor-managed MongoDB Atlas option. The company is taking a similar view on managed Kubernetes.

That doesn’t mean engineers don’t push for new tools to be brought in to the ecosystem. “Sometimes you have to say no,” Hubin said. “Recently, we had people trying to bring in a new database. They had a point, but if the standard option is not quite as good, it is [still] overall better for the company to control the number of databases we use.”

Every large organization has a broad cohort of engineers, some who focus on building resilient systems and who deliver features to customers at velocity, and others who desperately want to tinker with the latest technology. Both have value, but they need to be managed carefully, Two Sigma’s Fournier said.

[Also on InfoWorld: How Docker broke in half]

“You want people who are excited to look under the hood and understand new things—because I need people to manage bare-metal Kubernetes—but I also need people who are excited about looking into new things, understanding how it works, identifying where it can be useful across the firm—good partners to prototype with and determining whether it is worth investing in unlocking it,” she said.

How vendors are contending with complexity

Like many of his peers in the cloud software business, Kelsey Hightower, principal developer advocate at Google Cloud, sees the current level of choice available to developers as both a “gift and a curse.”

The gift is the availability of a near-limitless catalog of technology to build with. The curse is that developers “are being pulled into situations where we leak the infrastructure up into their workflow.” Now, with many vendors focusing on managed services and abstraction, the pendulum appears to be swinging the other way. After the great fragmentation, are we due a great consolidation?

“There is more to this profession than writing code; that is the means to an end,” Hightower said. “Maybe we are saying we have built enough and can pause on building new things, to mature what we have and go back to our respective roles of consuming technology. Maybe this is the happy ending of the devops and collaboration movement we have seen over the past decade.”

The market is responding to this complexity with an ever-growing list of opinionated services, managed options, frameworks, libraries, and platforms to help developers contend with the complexity of their environment.

“No vendor is or will be in a position to provide every necessary piece, of course. Even AWS, with the most diverse application portfolio and historically unprecedented release cadence, can’t meet every developer need and can’t own every relevant developer community,” O’Grady wrote in a 2020 blog post.

That being said, “there is ample evidence to suggest that we’re drifting away from sending buyers and developers alike out into a maze of aisles, burdening them with the task of picking primitives and assembling from scratch. If the first era of the cloud is defined by primitives, its days are coming to an end. The next is likely to be defined by, as the computing industry has since its inception, the abstractions we build on top of those primitives,” O’Grady wrote, in a different post.

While assembling these primitives into coherent internal platforms has already proved something of a successful workaround for many engineering-led organizations, more traditional enterprises are absolutely going to look to their providers to help them ease this complexity.

How to choose a low-code development platform