LOW CODE VS NO CODE IN THE ENTERPRISE

Not many groups have to be as adaptable as enterprise software developers. In the fast-moving world of software development, every day as an education. Today's enterprise coders are most certainly familiar with DevOps, at least peripherally. While some find the DevOps philosophy has rigid expectations, others believe that is changing.

One example of this perceived rigidity is exercised in the idea that DevOps must remain hand coded, just because that's what is traditionally done. This leaves little room for low code or no code. These digital schools of thought, while all intended to foster an environment of efficiency and cooperation, seem to be at philosophical odds. But that may not be true much longer.

Here's a look at the current issues in DevOps surrounding low code and no code initiatives.

Implications of DevOps for Coding

With so many shifts in IT ideology over the years, it seems like new IT operational perspectives are as common as the change in winds. One thing that has remained constant at the core of IT organizations is agility.

Surely, it's easy to understand why agility is so important to IT, a discipline that requires enterprise organizations to stay up to date on new technology, innovation, training and methods. DevOps is an answer to the question, "How do we remain agile?" Here's a brief history of DevOps and how it relates to coding schools of thought:

Brief history of DevOps

In simple terms, DevOps is a way of thinking in IT that bridges the communication gap between development teams and operational ones. By fostering a better environment for communication, the software can be delivered more quickly.

This mindset came about at the turn of the century. It was an answer to a common problem many enterprises were facing: how to get fully developed code through production and shipped while ensuring all teams involved communicate well. The answer is creating a culture where each cog in the machine understands their part of the overall mission and how they work together.

In one State of DevOps report, Puppet Labs stated that enterprise organization fail exponentially less and recover exponentially faster when DevOps is properly implemented throughout the organization. The groundwork for implementing DevOps is simple, when you have the right tools within an organization and the right culture to implement them successfully from a holistic worldview, you'll soar.

From a 30,000-foot view perspective, that all sounds great. But what does that mean for development teams who are at the heart of a DevOps operation?

DevOps & Low Code or No Code: friends or foe?

One of the more significant movements in IT in recent years is a trend toward low code and no code tools and platforms, and it isn't going away. Each will be defined in the section below:

- **Low code** is a development movement where time-consuming manual processes are automated, without hand coding, using a visual IDE environment, an automation that connects to backends and some kind of application lifestyle management system. The whole unit is called a "low code platform".
- **No code** platforms, similar to low code platforms, use a visual application system that allows users to build apps without coding. Usually, this includes drag and drop processes. One example of this is Salesforce CRM, which allows people with coding skills to code, and those who don't have those skills can create simple applications without using any code at all.

Why are these important? Well, that's a more complex answer. Like DevOps, Low Code and No Code platforms are designed to increase the agility and effectiveness of organizations. But unlike DevOps, they don't traditional require hand coding. That leaves many IT organizations wondering what role, if any, low code and no code platforms could play in their DevOps organization.

After all, DevOps organizations are synonymous with having skilled developers on hand, and now being introduced are platforms where developers don't need to have skill at all. What does that mean?

Low code and no code models, which were traditionally designed for lay-consumers, are now being fashioned with enterprise needs in mind. This is a natural progression since the disruptive rise of SaaS in enterprise business and DevOps, to boot. There's a laser focus on agility, which has created an enterprise market for low code and no code services.

Add with that, low code and no code platforms are becoming more disruptive in and of themselves to meet enterprise needs. Forbes describes how low code and no code platforms are incorporating innovation like AI to help them further automate enterprise functions. This is sure to help these

platforms cast a wider customer net.

Ultimately, while they didn't start out this way, low or no code platforms and DevOps are good bedfellows. They both move enterprise organizations toward the same result: greater agility. At one time, low code and no code platform lacked much of the functionality required to make them useful tools in a DevOps environment but, certainly, that's changing.

Low code vs no code: How to choose

In this article, we've been using low code and no code, together, when referring to how this innovative platform technology will change the face of DevOps, but if you're considering one for your organization, how do you know which one to choose and when?

Here's a glance at low code and no code for your organization:

bmc

Low Code	No Code
Use for more complex applications	Use for reporting, analytics and tracking apps
Usually for apps that are foundational or run important processes for a business	Apps that evolve with frequent updates and changes in use-case
Apps with: • More than 5 years lifecycle • Fewer updates	Can be integrative or stand alone
Can be mission critical	Good for self-deploying apps
Offers more developer control	Mobile responsive

Low Code

- Use for more complex applications
- Usually for apps that are foundational or run important processes for a business
- Apps with more than 5 years lifecycle, and fewer updates
- Can be mission critical
- Offers more developer control

No Code

- Use for reporting, analytics and tracking apps
- Apps that evolve with frequent updates and changes in use-case

- Can be integrative or stand alone
- Good for self-deploying apps
- Mobile responsive

It's likely that most organizations will have uses for both low code and no code applications. Between both, a lot of development ground is covered, but is that enough to push out hand-coding completely? Many people think so.

The end of an era

We've established that not only are low code and no code environments great for DevOps, but they are on the rise. What then, does that mean for the future of skilled hand-coding and development in DevOps organizations?

Most experts, ourselves included, believe that hand coding will eventually go the way of the dinosaur. Mostly. There will likely always be business use-cases for hand coding that are specific to a particular enterprise. With low code and no code, you basically pick and choose from a robust range of presets and templates, tying it together with only small snippets of code or none at all. While convenient and agile, it's unlikely these platforms will cover all the specific needs of every enterprise, so it's likely many businesses will need to run a low code and no code environment in tandem with a hand code one.

Additional resources

For related reading, explore these articles:

- BMC DevOps Blog
- Infrastructure as Code (IaC): An Introduction
- Jobs-as-Code: The Power of Shifting Left
- What Is Spaghetti Code? (And Why You Should Avoid It)