How AI Affects Low-Code/No-Code for DevOps

TechTarget

In this handbook:

How AI Affects Low-Code/No-Code for DevOps

TOM NOLLE, FOUNDER AND PRINCIPAL ANALYST

Software development is a complicated task for trained professionals, right? Well, there's a problem in the development world where the need for software has far outrun the number of software developers.

Most chief experience officers are familiar with citizen developers using low-code or no-code, and many also accept that AI can help improve the quality of these nonprofessional developers. Some have also learned that low-code and no-code can improve DevOps, and that AI can enhance their relationship too.

Let's examine the role that low-code and no-code play in DevOps, and the ways that teams can integrate AI with these approaches for software development.

Low-code and no-code in the DevOps environment

Low- and no-code tools are designed to democratize programming. They do this by introducing models or templates for various types of applications and providing simple mechanisms to fill in details without a heavy code input from developers.

The main difference between low-code and no-code tools is the extent to which they allow a complete escape from traditional programming languages. In turn, these tools have a limited scope of application development they can support.

From a DevOps perspective, low-code and no-code tools create a curious mixture of greater consistency and lower overall control with the code. Platforms for both low- and no-code development tend to generate code based on a specific model, and the overall testing and deployment considerations for this code are more model-specific than application-specific, particularly for no-code. As a result, the deployment processes are fairly standard, particularly for no-code.

When combined with AI, low-code and no-code development will have a similar focus in the DevOps environment, but it will have a broader effect. Regardless of whether the organization uses low-code or no-code, teams can get the most from AI when they focus on one of two approaches. The first approach is a low- or no-code

platform with integrated AI. The second is to select an AI-equipped application observability platform where teams can then fit specific DevOps tools.

The role of AI-assisted development

AI's role in low-/no-code pipeline optimization varies with the steps involved. Ultimately, the goal with low-code and no-code tools is to ensure that citizen developers don't have to rely on operations professionals or programmers to manage DevOps on their behalf. Most of the popular low- and no-code tools are really development platforms that include DevOps and Agile development features. AI is a recent addition to this capability set but isn't widely available yet.

Microsoft has promoted its Power Platform service with AI incorporated. Power Platform includes a low-code tool called Power Apps, a data visualization no-code tool, Power BI, and a workflow automation platform called Power Automate. Other low-/no-code platforms with integrated AI are Appian Platform, Mendix Assist and Pega.

Most of these platforms make use of copilot technology to add AI capabilities when they create programs or scripts. These tools are often easier to use and more powerful than those that generate code with little or no human intervention. Power Apps users find AI copilot features highly valuable. AI can also be useful in the no-

code Power BI, but since no-code tools are typically highly graphical or form-based, some users won't find AI as helpful there.

Using any platform tool for low-/no-code has limitations from a DevOps perspective since these tools don't fit as easily into an arbitrary DevOps workflow when compared to discrete tools. In no-code development, this problem is less likely to arise because professional developers and their tools and workflows are rarely used. Since low-code can be used by a development team for routine tasks, it's important to ensure that a low-code platform can integrate with the rest of the workflow elements.

Implementing AI in low-/no-code development

If an organization doesn't have a single platform for low-/no-code, an overall visibility-focused AI-equipped operations platform is a good strategy. Most no-code and some low-code applications run without the IT ops team's supervision and scheduling. This means that observability at both the application and resource level is more important in optimizing costs and quality of experience. Generalized AI-equipped platforms like Dynatrace provide good overall application/resource visibility. It's also possible to add AI tools to platforms like Prometheus.

Generally, both low- and no-code technologies will use either a component drag-and-drop model, a tabular/form model or a wizard/assistant model. Low code provides greater agility and flexibility because it allows developers to create components with

traditional programming tools. These components can then be added to the already-assembled inventory of things to create an application.

No-code may allow limited customization options, but it relies almost totally on generated elements. In low-code applications, developers tend to prefer a copilot or assistant model that suggests code. Tools like GitHub's Copilot can serve both as a low-code aid and a general developer tool. It can be argued that coding assisted by generative AI (GenAI) is itself a low-code approach.

In no-code, the current trend is toward AI-based generation of visualizations and documents rather than augmentation of traditional no-code techniques. The AI effect in this field is a push toward *output-centricity*, so that no-code with AI becomes a data analysis strategy more than a development strategy.

Just as GenAI coding is a form of low-code, many of the AI/ML applications available in the cloud or as discrete software are forms of either low- or no-code. Akkio, Amazon SageMaker, Apple Create ML, Google AutoML and a host of evolving GPT-based tools can generate charts and tables from data, which resembles an AI-evolved form of no-code.

The cloud-hosted AI-centric approaches to low- and no-code have no real deployment associated with them; think of them as low- and no-code-as-a-service. The amount of time, money and commitment focused on this approach is by far the

largest in low-/no-code, so it seems likely to dominate in the future. This approach trades the traditional application-building models for AI, so users actually build an AI application rather than coding in any way.

DevOps is the workflow that links development to deployment. Low- and no-code technologies support citizen developers and DevOps professionals who build basic applications. AI plays a significant role in that development today and will surely play a larger role in the future. AI may also enhance the testing, deployment and management tasks in a DevOps flow as well. The best approach to incorporating AI in any of these missions is to use tools that integrate AI rather than to attempt to use AI independently.