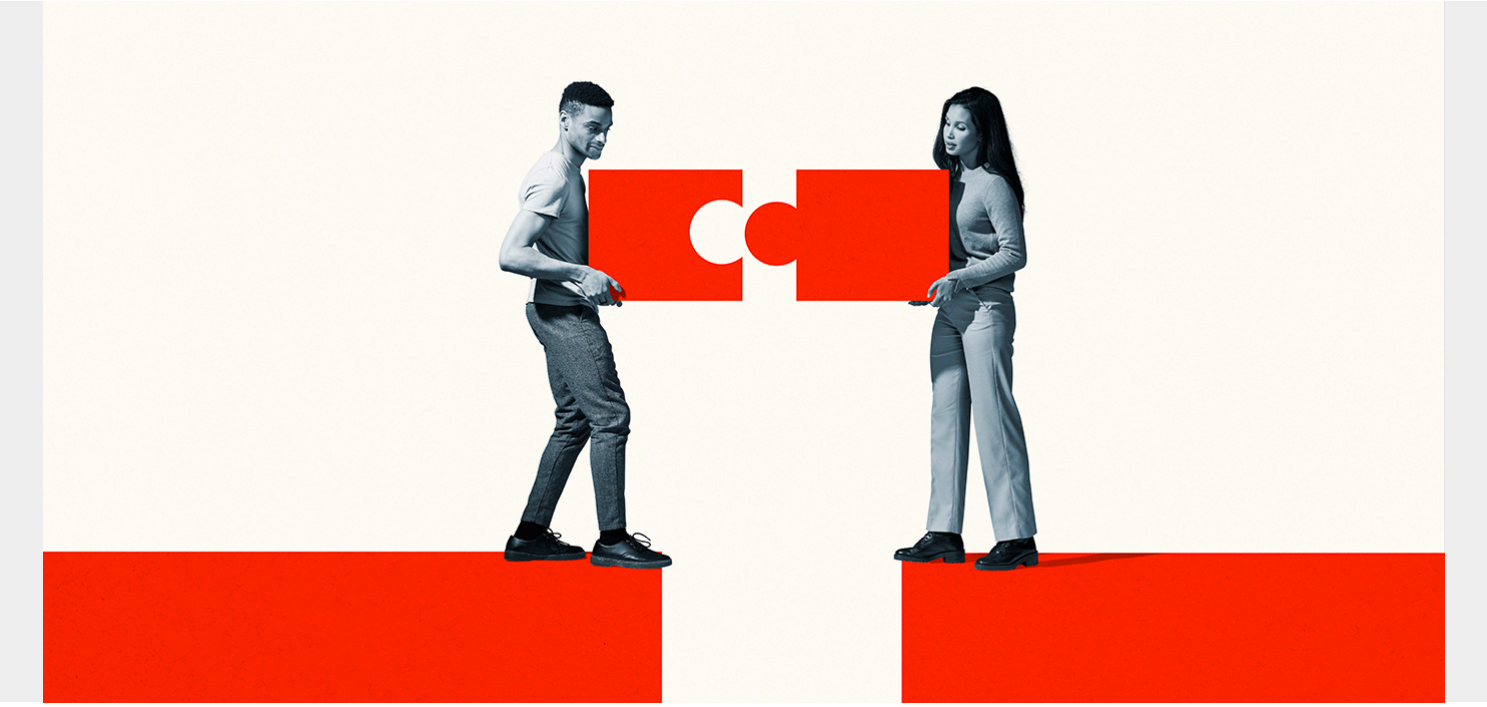
MAINFRAME AND DISTRIBUTED: UNITING AN IT HOUSE DIVIDED



Historically, there has been a divide between mainframe and distributed teams in large organizations. In recent years, many organizations have seen significant increases in the quality, velocity, and efficiency of their software delivery thanks to the adoption of Agile development and DevOps. But while progress has been made, there are still organizations seeking to maintain the status quo, continuing on with an IT house that is divided. This approach is a mistake.

Why Mainframe and Distributed Can't Be a House Divided

DevOps calls for the breaking down of silos, for the unification of development and operations, and not just for one system but across all systems, mainframe and distributed. In DevOps, everything functions in unison at the same speed in order to improve the quantity and quality of functionality delivered.

Let this serve as a challenge to IT organizations that are still advocating for the division between mainframe and distributed or allowing that division to persist out of apathy, because a house divided cannot stand. As more applications span platforms, having mainframe and distributed running at different speeds—slow on the mainframe, fast on distributed systems—is only going to damage the customer experience your organization is trying to deliver through digital means.

The challenge, which is an exciting one, is to open the mainframe. Opening the mainframe means implementing the same Agile and DevOps best practices non-mainframe teams are likely already using—iterative development, earlier testing, Continuous Integration/Continuous Delivery, and so on.

Instead of upholding the divide between mainframe and distributed, we need to become one thing,

and I'm betting joining the "fast" side of the house will beat being on the "slow" side every time.

Innovating for Mainframe and Distributed DevOps

Mainframe and distributed should be on the same level, but you need a way to get the mainframe there. For a better idea of how to do this, I would read "Ten Steps to True Mainframe Agility," a BMC AMI DevX E-book that lays out a flexible plan involving many of the tools, processes and performance milestones you need to transform your mainframe. It's useful even if you have started the Agile and DevOps journey, as there is always room for improvement.

The bulk of data and processing remains on the mainframe. It is the most efficient platform and the applications are right there with decades of business knowledge built in. What it still lacks in many cases—not due to the platform but to poor practices and tools—are automation, visibility and integrations.

Rather than opting for the status quo, mainframe applications and tools must fit seamlessly into DevOps. They must adapt and become more open and flexible through REST APIs. Dashboards and management tools must see all of development, not just one side of it, and development must move from Waterfall to Agile, working with one IDE.

Utilization of a mainframe-inclusive DevOps toolchain is the way of the future. Just as organizations must change their approach to the mainframe and its role in enterprise DevOps, the platform and the developers who work on it must adapt. Be on the side of change. Build one strong house for mainframe and distributed instead of fighting to maintain two.