TREXIN INSIGHT PAPER

October 5, 2022

IMPLEMENTING AZURE DEVOPS

What to consider when moving from a current requirements workflow tool to Azure DevOps.

This Trexin Insight Paper (TIP) will give your team an explanation of the Microsoft Azure DevOps Suite and the considerations to keep in mind as you implement Azure DevOps as your primary requirements management and workflow tool.

Many companies start using Azure DevOps to gain efficiencies from its robust Continuous Integration - Continuous Development (CI-CD) pipeline functionality. As a DevOps organization matures into tool usage and realizes the benefits from their core development teams' usage, CIOs often see the strategic benefits of extending into additional Azure DevOps functionality. Using one tool, built for the needs of multiple team roles to define, track, test, collaborate, and manage their software delivery and/or their Agile management of business project flow, has a strong allure.

At the management level, this adoption makes good sense strategically, tactically, and financially, and can even increase user satisfaction if implemented properly. The use of a flexible, customizable system built on Agile Principles allows for easy support from senior management, compared to existing Software Development Life Cycle (SDLC) tools that were historically pieced together from too many different solutions which are now hosted on custom servers, enterprise, and vended cloud, and made for less flexible ways of working.

For organizations making the leap from waterfall-based development to Agile and CI-CD modern delivery, the potential of having a single tool to manage end-to-end discovery, requirements activities thru test, and deployment activities makes Azure DevOps a top contender. Like many strategic tooling choices, the buy decision is the easy part. We recommend that every organization needs to consider the Azure DevOps usage, configuration, and adoption process in multiple contexts to be set up for success.

UNDERSTAND AZURE DEVOPS COMPONENTS IN THE CONTEXT OF DEVOPS AND SDLC NEEDS

First, make sure your teams align to an understanding of which parts of the overall Azure DevOps suite are being delivered. What each individual team member means when they talk about Azure DevOps can mean different things to different developers, product owners, and the other team members who drive value for your organization.

There are multiple parts and licensing considerations for each component in Azure DevOps. The point of view of the team that brings the tool to the organization will impact expectations.

The product suite includes the following tools:

- **Azure Boards** is where the planning and tracking development and implementation work is done. With Kanban-style boards to track work, backlogs for sprint planning, and more, Azure Boards is a valuable tool for the whole team, including managing day-to-day business operations.
- **Azure Test Plans** is used for manual and continuous testing and allows the team to collaborate while integrating analytics and data collection.
- **Azure Repos** provides Git repositories for controlling your code and tracks code changes over time, allowing for coordinated code changes and access to change history.

- **Azure Pipelines** provides build and release services for support of continuous integration and delivery of your applications.
- **Azure Artifacts** supports package sharing into your pipelines.

These products were created to work together seamlessly, allowing your company to focus less on organizational challenges related to handoff between tools and more on providing valuable deliverables to your customers.

UNDERSTAND USER TYPE AND USER READINESS IN CONTEXT TO TOOL BENEFITS

Before you release a tool central to daily project execution, you must know and understand your target population of users. Azure DevOps can be rolled out enterprise-wide, divisionally, or departmentally, depending on the desired usage and scale for the company IT strategy. Azure DevOps can be for IT implementations only or can have a broader business context or technical support use.

To begin, determine both your individual teams' and your enterprise's background with project execution tools. Teams may come from a Jira background or similar, whether that is having used ticketing or general progress tracking or being power users of all of Jira's IT development tools. Teams with deeper development tool backgrounds such as Integrated Development Environments (IDEs), build automation, test automation, or knowledge of Microsoft development products may be able to pick up Azure DevOps faster. Those same teams may be comfortable with their existing tool and may want Azure DevOps to be configured to look like the old one. Evaluate those needs and requests carefully. The robustness of the Azure DevOps suite requires the tool be used as designed or in specific way to reap the benefits of the downstream automations.

On the other hand, the personnel adopting a tool like Azure DevOps may have never used a similar tool before. The business user context of Azure DevOps requires a mindset shift and understanding of one or more variations of Agile or Capability Maturity Model Integration (CMMI). Without training and support to learn these common and seemingly interchangeable methodologies, many users can become frustrated with having to learn a new tool and a new way of working.

Even the leap for an experienced ticketing workflow system user or for team members with prior Agile experience outside of a tool like Azure DevOps can create frustration, resistance, and need for relearning support. By the very nature of a form, field, and state-based workflow, users must enter and feed specific fields to support the benefits that come from the tool. Agile is a flexible framework. Agile tools are often not. Expect a short-term decrease in productivity in all teams and strategize adoption plans by appropriate grouping to maximize productivity and return to previous and faster deployment gains.

Overall, it is important to remember that good tool selection does not necessarily lead to a fast and simple adoption process. When you consider that Azure DevOps also supports the needs of multiple types of development, Agile functionaries, and product management roles that enables the end-to-end needs of software delivery lifecycle (and can support business process surrounding SDLC), it is important to invest in tool training, usage patterns, and most importantly understanding who the users are who will be supplying and receiving the benefits of the various components of the tool first to ensure your strategic investment pays off as desired.

If you and your team have questions regarding Azure DevOps or are ready to make the leap into the tooling suite, reach out to a Trexin Advisor for further information.

Stay tuned for more TIPs on the adoption of Azure DevOps.



This TIP was written by Elizabeth Badrov, Katie Laurence, and Madeline Hultquist. Elizabeth, Katie, and Madeline welcome comments and discussion on this topic and can be reached at elizabeth.badrov@trexin.com, katie.laurence@trexin.com, and madeline.hultquist@trexin.com.