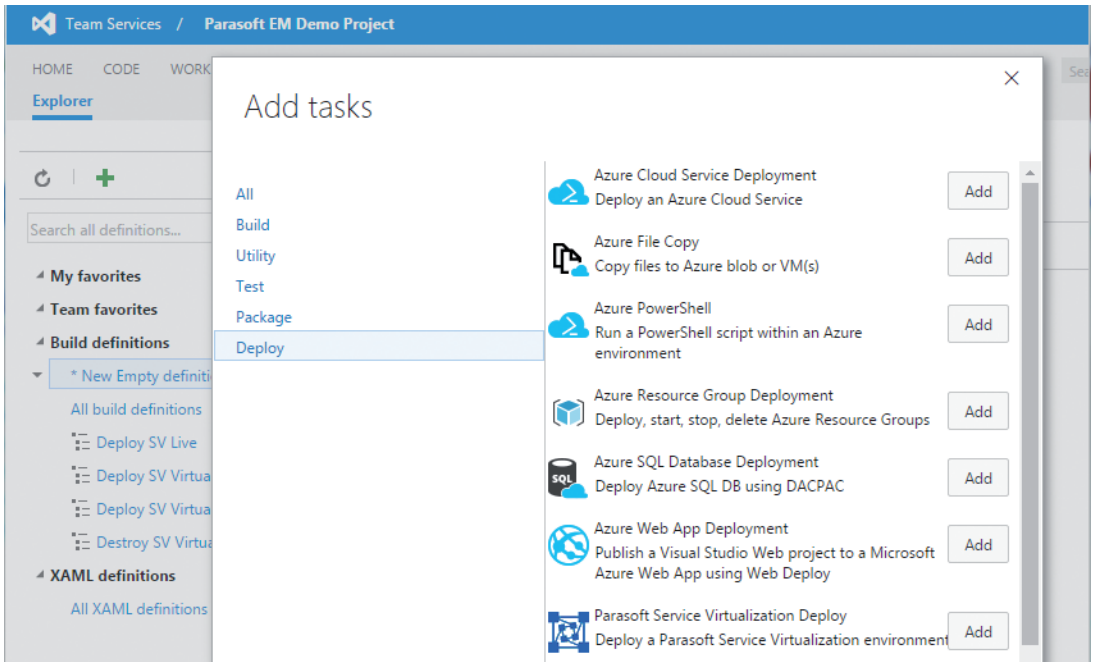
Service Virtualization in a Microsoft Environment—Scaled by Microsoft Azure

With technologies like Microsoft® Azure™ (for elastic scalability), Microsoft® Visual Studio Team Services™ (for build and deployment automation), and Parasoft® Service Virtualization™ (to simulate and access dependencies that are beyond your control, still evolving, or too complex to configure in a test lab), there's no reason why a complete, realistic test environment should be out of reach. By taking advantage of a Microsoft ecosystem with VSTS and Azure, organizations gain immediate access to scale and bandwidth. This provides the resources needed to enable flexible and ubiquitous access to application stacks that are under your control for DevTest purposes. But what about the dependent system components that are beyond your scope or control (e.g., third-party applications, SAP, mainframes, not-yet-implemented services, etc.)? You can simulate their behavior using Parasoft Service Virtualization—eliminating the final test environment access gap that commonly impedes teams' testing efforts.



Dell recently gained first-hand experience in how using service virtualization in a Microsoft environment is key for rapidly deploying complete test environments on demand. The enterprise software division's leaders were confident that thoroughly testing each user story as soon as it was completed would help them achieve their

goals of a) delivering more business value in each of their two-week Agile sprints and b) finding defects earlier in the SDLC. However, limited access to test environments that accurately represented their disparate systems made it difficult to exercise the required end-to-end transactions as early and often as needed.

By using Parasoft Service Virtualization with Microsoft Azure, they can now simultaneously spin up the various test environments required to test each user story from the moment it's completed. This enables them to quickly expose and address problems with new functionality before the end of each Agile sprint—as well as to continuously execute a robust regression suite that alerts them immediately if code changes break existing functionality.

The Challenge: Rapid Access to Realistic Test Environments

Virtually all of the 200 applications that this group is responsible for have a complex set of dependencies, including direct interaction with the company's multibillion dollar sales portal. Accurately testing realistic end-to-end transactions across these applications requires access to a complete test environment, including many different internal and external systems communicating with an assortment of protocols and message formats.

Further complicating matters, their test plans require that each user story be tested versus dependencies that are configured with extremely specific test data, response logic, and performance conditions—and configuring the environment for one user story means disrupting the configuration for the others. Topping it off, now that the company performs parallel development, tests sometimes involve services that are not yet implemented or are still evolving during the current iteration.

The Solution: Service Virtualization Simulates Environment within Azure

The organization knew that Azure DevTest environments could help them provide immediate access to the application stacks which could be logically imaged (as long as they had configuration control). However, they still needed another strategy to provide access to the functionality of the continuously-evolving sales portal and the hundreds of other services and third-party applications beyond their control for traditional server virtualization.

Using Parasoft Service Virtualization, the DevOps team built a core library of 500+ service virtualization assets that provide flexible, on-demand access to the behavior of the many dependencies required to exercise realistic end-to-end transactions. These virtual assets were built by capturing actual behavior, and are architected so that they can be easily updated to reflect changes in the dependencies—as well as extended to cover any additional conditions that need to be simulated for testing a particular user story. For example, response logic, data, and performance conditions can all be reconfigured instantly, even by a novice user.

With this foundation set, the next step was to establish an infrastructure that would automatically spin up complete Azure DevTest environments tailored for testing each user story. Since the infrastructure enables multiple test environments to be active simultaneously—with any given test environment having zero impact on the other test environments simultaneously being used to validate other requirements—they were termed "bubbles." These bubbles are automatically provisioned whenever they're needed, then destroyed as soon as the associated testing is completed. In other words, they are dynamic yet disposable.

Now that the team can immediately and continuously test each user story against a complete, realistic test environment, their sprints are not only more predictable (fewer carryover user stories), but also more productive (increased velocity). Most importantly, this increased delivery speed does not require any trade-off at the expense of quality. In fact, quality has actually improved. Previously, only 30% of their defects were being discovered within the delivery pipeline. Now, the vast majority of defects are identified and resolved before the user story is deemed "done done."

PARASOFT

USA PARASOFT HEADQUARTERS / 101 E. Huntington Drive, Monrovia, CA 91016
Phone: (888) 305-0041 / Email: info@parasoft.com

© Parasoft Corporation All rights reserved. Parasoft and all Parasoft products and services listed within are trademarks or registered trademarks of Parasoft Corporation.
All other products, services, and companies are trademarks, registered trademarks, or servicemarks of their respective holders in the US and/or other countries.