Case study

The right environment for the right test: A missed opportunity

A customer in the manufacturing industry that manufactures recreational vehicles is implementing Dynamics 365 for Finance and Operations and has created a testing strategy that included four testing cycles, including user acceptance testing, as part of their plan.

The team included the most common test types except for performance testing under the assumption that the first rollout would not require it since the extension of the system was low, and the customer will be implementing just manufacturing business unit. Next rollouts will include the servicing business, including the warranty management of recreational vehicles plus other operations. The servicing operations will continue to be executed as a third-party solution for now and their dealership network will be using this system to order parts and honor warranties.

Very soon, during the first testing cycle, the team started to test across the first wave of the solution where there were no integrations, so completing testing was satisfactory in terms of the cycle performance.

As the team got ready for their second wave of testing, they started to introduce some of the integrations as part of the scope for the test cycle, but those integrations were emulated with dummy data and the team introduced some migrated data. Integrations worked for their purpose and the volume of migrated data was small.

For their third testing cycle, things looked great and they were ready to do a more comprehensive testing by using end-to-end testing, this time with real integrations and real volumes of data migration.

At combining the testing for having a larger volume of users for the test and integrations running with similar volume of operations to what they expected in production, the team hit the first major blocker. The solution was not keeping up with the needs of the test cycle, and they were not even in production. The first reaction from the team was an underperforming infrastructure, which raised the concerns of the business stakeholders upon learning the test cycle outcome.

The team was ready to prepare for UAT and decided to continue expecting that this would not be an issue in production due to it having higher specs. They assumed the production environment would be able to solve this performance challenge, so they decided to continue, complete UAT, and move to production. The customer signed off and preparation moved to the next stage to get ready for go live.

The big day came and production and all the departments switched to new system. The first day was normal and everything was working great. The team decided to turn on integrations for the servicing solution on the second day. When the second day came, they were ready to go interconnect with the service department, and integrations started to flow into Dynamics 365. This is when they had their first business stopper: they had a sudden decrease in performance, users were not able to transact in Dynamics 365, service departments from the dealerships were not able to connect effectively, shipments started to slowdown, and the shop floor was having a hard time to trying to keep inventory moving to production.

The implementation team was providing hyper-focused care and immediately started to check for what could be blocking the operation and they found the problem. The integration was creating a bottleneck by making important OData calls to check inventory, creating sales orders, and invoicing while connected to the dealership's partners using the servicing solution. They stopped the integration and things went back to normal, but it was clear that the integration was the cause.

The team found a solution after troubleshooting, but it will require a change to the integration design pattern and fast movement to solve it and keep operating. Part of the revenue from the recreational vehicle business comes from servicing and providing parts to the dealerships. It took days of manual work to keep things working during this painful first week of go live, which impacted the perception of the organization in the solution.

In retrospect, validating designs for integrations with a volume of data was key to preventing this scenario and the issue they had during real operation. The team's morale was affected after working hard during the implementation season, all because they missed one important test—the performance test. The team thought emulating early and validating in production would be OK, but the cost of that decision was high.

This team learned that including performance testing as early as possible, with actual volumes in the right environments that perform at the same volumes expected in production, would help to detect a design problem and correct it in time. The team also learned that "testing" in production for the first time, and with the real volumes, was not the best idea since the time available to make any fixes was reduced due to the live operations and correcting as they go introduced bigger risks. The assumption of the team that they were implementing almost standard functionality led them to think that the solution would perform fine, but the actual customer challenge made the solution unique which requires extensive testing.

For the next phases, this customer included performance testing. They dedicated an environment to stress test the solution with their own business needs and they executed these tests earlier with additional scenarios and UAT that included parallel processing of people testing and integrations running. They were able to have second go live to include their accessories manufacturing business and it was a breeze in comparison.

It is never too early to start testing and do so according to the project needs. The later the testing, the bigger the risk.