COMMON PITFALLS: ALIGNING IV&V AND TEST WITH AGILE DEVELOPMENT

TestPros Whitepaper

Agile Development without aligned IV&V & Testing isn’t very Agile

The Agile Software Alliance\(^1\) defines Agile Software Development as:

*Agile Software Development is an umbrella term for a set of methods and practices based on the values and principles expressed in the Agile Manifesto\(^2\).*

Solutions evolve through collaboration between self-organizing, cross-functional teams utilizing the appropriate practices for their context.

This whitepaper explores the common pitfalls we witness and help to avoid as we support the move to Agile development, continuous integration, continuous deployment, and DevOps. TestPros, an independent IT assessment firm established in 1988, has supported Agile development lifecycles since the inception of the approach. This document addresses common challenges related to the concept of establishing the right self-organizing, cross-functional teams and implementing appropriate practices for testing across those teams.

TestPros re-engineered a client’s existing Testing approach to keep pace in their Agile development lifecycle by using an efficient keyword-driven framework.

This resulted in an 800% improvement in performance measured in test execution time while establishing 100% test coverage.

---

1— https://www.agilealliance.org/
2— https://www.agilealliance.org/agile101/the-agile-manifesto/

---

The Pitfalls to Avoid

- Believing Agile Developers are Fully Competent Testers
- Treating Testers as a Commodity
- Believing Automated Testing is the Same as Agile Testing
- Planning to Save Money Immediately using Test Automation/Frameworks
- Fragmented Development and Testing Standards
The Common Pitfalls

Pitfall #1 – Believing Agile Developers are Fully Competent Testers

… or Database Administrators. or System Administrators. or Configuration Managers … we can go on with other specialists such as Project Managers, Schedulers, and others who contribute to the team, but the point is that Software Developers are not the only role where responsibilities and operations practices change in an Agile environment.

A key issue we observe is that many organizations rely on their developers to conduct the majority of testing, eschewing independent testing support. Good testers are generally good developers with yet another set of skills layered on top, including expertise in many of the excellent proprietary and open-source testing tools available today. Developers generally do not have the time to become expert in both their primary duties and the testing tools and related best practices. The result is a low percentage of automated testing, and an over reliance on risk-based testing because of limited testing resources.

Solution: Recognize that Testing is a specific skill set, and engage the right independent test team to perform integration, performance, and other technically challenging test types in parallel, while embedding them as part of your cross-functional Agile development teams to allow for early planning of testing resources.

Pitfall #2 – Treating Testers as a Commodity

Recognizing that good testers are generally good developers with yet another set of skills layered on top of that, why would their skill set be any

What Agile Development Means to Us

The Agile Manifesto is nice, but what does Agile Development really mean in our context?

Our basic concept of Agile Development:

Teams of software developers and other supporting staff working on multiple iterations, or versions, of software in parallel to reduce development cycles and get the high quality software functions the end user wants, or are statutory requirements, delivered as quickly as possible.

The iterative nature of Agile allows development to begin without all requirements being locked down, involving the end users to evolve requirements across iterations.

Pitfall #3 – Believing Automated Testing is the Same as Agile Testing

Developing automated tests takes time and expertise, whether using a proprietary tool set from leading vendors such as Hewlett Packard and IBM, or open source tools such as Selenium and Grindr. Automated tests unto themselves do save time in regression testing cycles, but in a fast paced Agile development environment where requirements are more
prone to change than in a traditional waterfall environment, it doesn’t mean testing will keep up with the development cycles.

**Solution:** Test Automation Frameworks. Test Automation Frameworks represent an approach where you develop automated test scripts such that the input parameters are variables versus hard coded. Instead of writing a new script for each code change, less experienced testers can literally fill out what looks like a spreadsheet with new input parameters that drive automated test execution. While it takes more time, effort and expertise to develop a solid test automation framework, and to develop the initial scripts, it pays off handsomely. In most environments, the result is a significant decrease in testing cycle time accompanied by greater test coverage, lessening the extent of reliance on risk-based testing where only critical functions are tested due to time constraints.

less differentiated than that of developers? The answer is that for the types of testers needed in an Agile environment, they are not. Using skilled testers is a must to keep up with accelerated release cycles, but many are reticent to spend money on testing subject matter experts (SMEs).

**Solution:** Plan and pay for talent commensurate with the testing requirements, especially early in the system development life cycle when developing the test architecture and Test Automation Frameworks. Your long-term testing cost will be less, and product quality higher.

**Pitfall #4 – Planning to Save Money Too Soon using Test Automation/Frameworks**

Developing Test Automation Frameworks represents an investment, not a quick fix to budget issues. In all likelihood, the initial costs will be higher than they would have been doing automated testing alone. The payoff comes in test cycles after the frameworks have been developed, when less senior testing SMEs are needed, and both more junior testers and developers can work with the variable-based scripts. You save money, but you also greatly expand test coverage.

**Solution:** Budget for a mid- to long-term ROI, not immediate. Use testing SMEs to develop solid frameworks that will pay off throughout the rest of the software development and maintenance lifecycle. Please see the figure above for a more realistic year to year cost estimate, keeping in mind that Test Automation Frameworks buy a lot more than direct cost savings.

**Corollary:** Avoiding testing frameworks because of the “risk” that costs won’t really come down. **Solution:** Check references at companies that have made the investment - Test Automation Frameworks deliver!
Pitfall #5 – Fragmented Development and Testing Standards

Many organizations perform software development in silos, whether the result of how projects are funded or because of historical structures in organizations resistant to change. The result is often competing toolsets (testing, CI/CD), and insufficient funding within each silo/contract/task order to support developing the initial test automation framework.

Solution: Standardize on tools, and build a test automation framework as soon as possible in the software development life cycle. If you are going to have multiple teams supporting development and testing, make the first testing task/task order be to develop a framework for broad application across the organization.

Don’t go cheap (or “Low Price Technically Acceptable/LPTA” for government folks) for framework development. There is a wide variance in testing SME skill levels, and hiring highly competent staff will pay off in the end with more effective and efficient framework designs.

In conclusion, Testing is a critical path item, and as such deserves just as much attention as other parts of the Software Development Life Cycle. Invest in the right people and processes from the start to help capture the many gains available from Agile development.

TestPros’ Supporting Services

Tools alone do not generate the efficiency required to keep up with modern Agile and hybrid development environments. TestPros helps its client establish Testing Centers of Excellence (TCoE), combining best practices and current technologies to achieve maximum efficiency and test coverage. This includes the development and application of sophisticated Test Automation Frameworks. The frameworks provide the basis for accelerating test automation—if there is a change to a test case, in many instances the tester only needs to update the test variables, in a spreadsheet or other form type, and does not need to update the test script itself. Scripts that drive the testing process and startup scripts do not need to change. The result is reduced cycle time and lower costs. After a TestPros Test Automation SME develops the initial test scripts, junior testers or developers can update the test case files, minimizing the need for expensive Testing SMEs over the full system development lifecycle. Yes, we often work ourselves out of a job! TestPros brings the required knowledge of the current processes and tools used to support the development of new test automation frameworks.