A DRIVER'S AND A DEVELOPER'S QUESTION – MANUAL OR AUTOMATIC?

Overview: Manual transmissions in cars have largely fallen by the wayside, replaced by the ease, convenience, and faster shifting of automatics. In a similar way, automated testing is becoming more commonplace because of the significant gains in speed, quality, and consistency it offers versus manual testing.

I am a "car guy," maybe it's because I'm from Detroit, but I've always loved cars. I like everything about them, especially driving them. When driving I prefer to use a "stick," a manual transmission. I like the connection I feel, the control. Learning how to drive a manual can be difficult at first, but soon it becomes "automatic" in that you don't realize you are doing it.

While this has become my preference, the car I regularly drive has an automatic transmission. Why? Well… At first automatics were limited, 2 speeds, and were "sloppy" having worse gas mileage. Over time they improved, adding speeds so that now the mileage they get is better. This has even made its way to the ultimate test, the racetrack, where the times automatics can get are better – they can shift faster than the human driving them. The only reason to drive a manual now is because you like to; there is no logical reason besides a lower initial cost.

I see the same trend taking place in software development. Many tests are still manual and, like driving a stick, they have become "automatic" to the tester and aren't really realized. The amount of time is not acknowledged, nor is the lack of testing because of the limitations of what can be done in the time available. But, as mainframe development teams move to Agile and shorter sprints, manual

testing won't hold up. Just like with auto racing, you need something faster than a human. It needs to be automated to improve quality and free developer time for design and coding.

There could be a feeling that automated testing isn't mature enough to challenge a good manual test. That feeling should be examined because things have improved for automated testing. Just take a look at Topaz for Total Test and Hiperstation. Both offer modern testing solutions that can be integrated into your DevOps toolchains so that testing can be triggered by an event rather than requiring extra developer effort. This allows testing to "shift left," meaning more frequent, consistent tests and better overall product quality.

When I've had a weekend car, it's always been a manual. I like the connection with the car and I don't have to deal with traffic or gas mileage. But when doing testing, I don't know of anyone that considers manual testing fun or a stress reliever. Testing is a "necessary evil" which, done manually, leads to inconsistent quality or, in a worst-case scenario, is avoided because of the time and effort involved. Automated testing reduces the developer's responsibility for this pain point, ensuring that consistent, thorough testing is performed throughout the software delivery lifecycle.

Automated testing may require a higher initial cost, but it allows developers the time to focus on the important things, with the payoff that you can complete development on time and with higher quality.