**Success with Test Automation**

**Bret Pettichord**

Over the past several years, tools that help programmers quickly create applications with graphical user interfaces have dramatically improved programmer productivity. This has increased the pressure on testers, who are often perceived as bottlenecks to the delivery of software products. Testers are being asked to test more and more code in less and less time. They need to dramatically improve their own productivity. Test automation is one way to do this.

By dedicating people on test automation, we have multiplied the other testers' productivity and have focused on the testing areas where the big wins are.

We have accepted "Good Enough" test automation. We've realized that just like with products we sell, software quality is a combination of reliability, features and timeliness. Understanding the importance of avoiding false positives has allowed us to make reasonable quality trade-offs.

We have tightened our testing cycle. Our testing is more consistent and repeatable. We are able to test on more configurations. We are constantly improving our test battery.

This paper describes several principles for test automation. These principles were used to develop a system of automated tests for a new family of client/server applications at BMC Software. This work identifies the major concerns when staffing test automation with testers, developers or contractors. It encourages applying standard software development processes to test automation. It identifies criteria for selecting appropriate tests to be automated and advantages of a Testcase Interpreter. It describes how cascading failures prevent unattended testing. It identifies the most serious bug that can affect test automation systems and describes ways to avoid it. It circumscribes reasonable limits on test automation goals.