

Quality considerations are of paramount importance in the software business. It has long been understood that the costs of defects rise significantly when they are not caught early. The agile community, for example, refers to “failing fast” as an important rallying cry and, as a result, sees continuous integration and testing as fundamental practices. Visual Studio 2010 provides an aggressive set of innovations around test and quality to help software teams deliver superior results.

Building Quality Applications with Visual Studio 2010

March 2010

Copyright © 2010 Microsoft Corporation. All rights reserved

Poor quality software costs the United States economy some $59 billion annually according to the National Institute of Standards and Technologies (2002)[[1]](#footnote-1). This money is lost through poor productivity and wasted resources. The study concludes that one-third of this cost could be saved by improving the testing infrastructure. In the last several years, according to the Standish Group’s Chaos Report (2009)[[2]](#footnote-2) shown in Figure 1, projects which are challenged or failed edged up again last year. There is no single silver bullet, but organizations can help themselves by using tools and processes that give themselves the best chance of avoiding project failures. At the end of the day, the only measure of quality that matters is whether or not the customer is happy with the final product. In concise terms this generally means that the product does what the customer wants (provides greater efficiency and effectiveness in their daily work) and it is free of defects which detract from the effectiveness of the software. Visual Studio 2010 is designed to give development teams the best chance possible of producing high quality software.

Figure 1: Chaos Reports show no significant improvement in project success.

# Meeting Customer Needs

Gathering complete customer requirements is difficult, no matter what tool you use, because much of this problem is rooted in often unpredictable customer interactions. But stability can be established and the needs of the customer can be monitored and quantitatively reported on as long as you have robust traceability from requirements to test cases to code to bugs and test results. A complete traceability story allows teams to determine which requirements are the highest priority, how much progress has been made on the requirements (work completed and remaining) and the *quality of those requirements*. Finishing a requirement is unimportant if the requirement has numerous defects associated with it because then it does not meet the needs of the customer using the general definition of quality. TFS provides just that degree of traceability and the data is easily discoverable through the fully customizable out-of-the-box reporting.

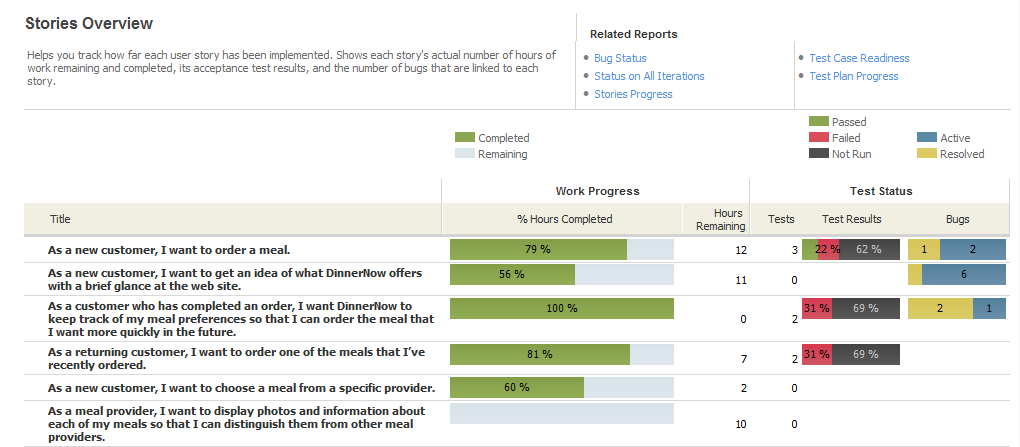


Figure 2 - Stories Overview Report.

At a glance the report shown in Figure 2 is fairly simple but it pulls together a multitude of data such as hours against requirements, test cases against requirements, test results for each requirement, and the number of active and resolved bugs for each requirement. At this point, knowing how well you have met the customer needs is no longer a guessing game – you have a definitive answer.

# 3 Steps to achieving Quality

There are really many steps involved in building quality applications but you can simplify the process down to just three key steps: Design, Develop, and Test.

## Quality Starts with Design…

Today, many architecture tools are separate from the development, requirements and testing tools. This makes it extremely difficult to architect systems so that the architecture is not discarded or ignored once coding begins. Developers have to constantly refer to other documents or other tools and eventually they just don’t. The side effect of this is that the design becomes out of sync with the actual code and traceability is lost at the technical level. This has an impact on the maintenance of existing systems as well. If you can’t understand the system, then maintaining it is difficult. Visual Studio 2010 Ultimate incorporates a new set of architecture tools built around UML which allow teams to not only architect solutions in their development environment, but link those designs to requirements in a way that they can be easily referred to and kept up-to-date. Figure 3 shows examples of the Use Case and Activity UML Diagrams, alongside another new architectural tool directly related to quality, the Layer Diagram.

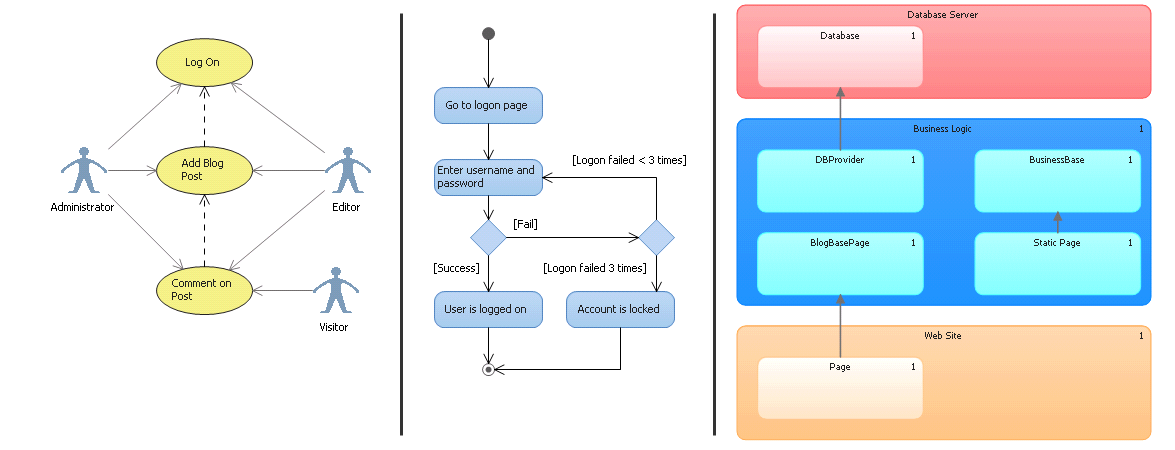


Figure 3 - A Use Case, Activity and Layer Diagram.

The Layer Diagram not only defines the layer design (a layer can be anything – method, class, namespace, assembly or an arbitrary grouping) and communication pathways but it can be used to constrain them. Once a design is created to achieve a particular goal, when that design is not followed the appropriate steps can be taken to either change the design or the code.

## continues with Developers…

Visual Studio 2010 includes many of the tools developers need to ensure they are meeting the needs of the customer. Features such as static code analysis (for .NET code and database code) provide insight into where there may be potential performance, stability, maintainability, scalability and security issues before they become a problem. Code metrics allow developers to determine where their code is most complex (a.k.a. less maintainable) and change the code before it becomes spaghetti. Developers can also perform comprehensive performance tests on their code, whether it is a WPF application or an ASP.NET application, to ensure the application meets the scalability needs of the customer. Figure 4 shows a Load test during the test run. Detailed information is provided on any aspect of any system involved in the testing. This ensures that enough information is gathered to determine where performance problems exist. You can also compare Load test runs to determine the amount of change in the performance characteristics over time.

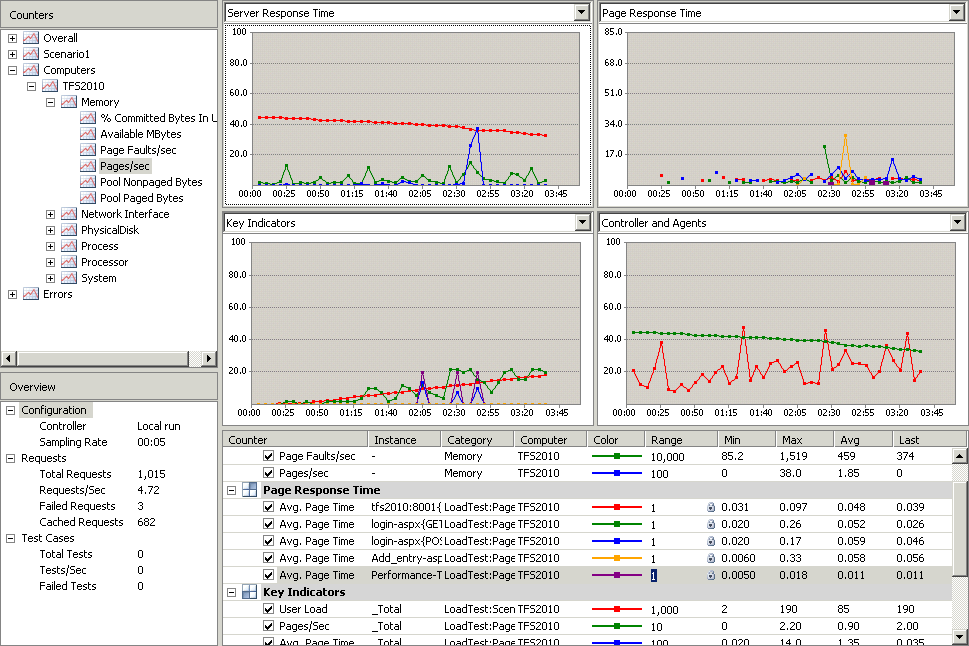


Figure 4 - A running load test.

After a Load test, if performance problems have been identified, detailed reports can be examined to determine exactly which parts of the application contain the problem code so that it can be fixed. Figure 5 shows an analysis of the user load against the application during the performance run. The figure shows not only the user load but the pages they were executing and any exceptions or other failures that occurred.

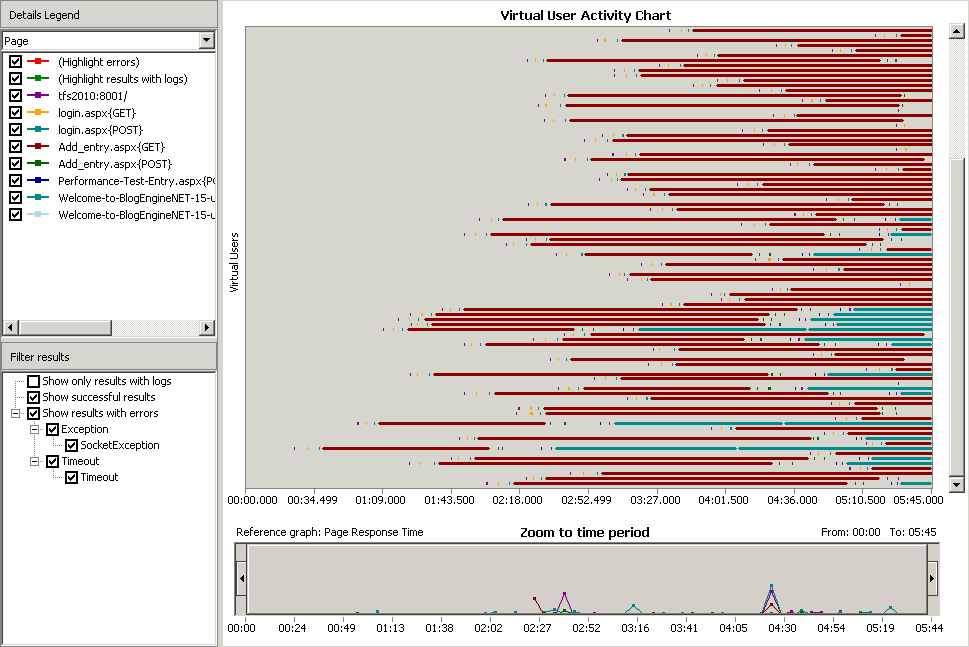


Figure 5 - Load Test Reporting.

Ensuring that defects do not make it to the final product requires that developers can find and fix bugs quickly during a debugging session. Microsoft’s latest innovation is IntelliTrace™ which allows for this and so much more. IntelliTrace™ allows developers to run an application, pause it at any point, and then move forward and backward in the debugging session to examine application state at a given point in time. It records a navigable history of the stack trace so that developers do not have to set breakpoints and hope they hit the right one! Not only that but when an exception occurs, developers can move immediately to the point where the exception occurred, determine why it occurred, and fix it. On top of this, testers running tests on the compiled application can record IntelliTrace™ logs so that when an error does occur the developer has access to everything that happened during the test run. This allows developers to find and fix bugs more efficiently than ever before.

Developers also have a responsibility to execute some tests before handing the code off to testers or even checking it into version control. Unit tests allow developers to discover if the code does not behave the way that it should at a granular level. For those teams that use unit testing heavily (such as Test Driven Development teams), the new Test Impact Analysis feature can reduce the number of tests which have to be executed and still give a high degree of confidence that the necessary tests have been executed. Test Impact Analysis analyzes code changes and correlates them with unit test runs to determine when a code change may impact the results of a previously executed test. This allows developers to spot check their code before sending it to the testers or the QA team.

Visual Studio 2010 also gives developers the ability to automate user interface testing. Developers can use Coded User Interface Tests to automate the testing of UI elements in their applications. Visual Studio 2010 automatically generates test code, either from a test run performed by a tester or from a run recorded by the developer themselves. This CodedUI test can be executed manually or incorporated as part of your build process to automate UI regression testing. Visual Studio Lab Management provides a virtualized testing environment which developers can use to run their code on clean systems which more accurately match supported release environments. This can help developers achieve their testing and experimentation goals in safe, easily replicated environments.

Automated builds have been a staple of Team Foundation Server and in TFS 2010 new features have made it easier than ever to discover coding defects before they ever make their way into the main code base. TFS 2010 sits on top of Windows Workflow 4.0 to provide powerful workflow capabilities with uses limited only by your imagination. One of the most innovative features that it is used for is called Gated Check-In and is a further refinement of Continuous Integration. With Continuous Integration, when the build breaks it’s already too late – the code stored in version control is already broken and the team has to take time out to fix it. Gated Check-In executes the build and any required tests *before the code is checked in*. This way, if the build breaks or the tests fail the code is never checked in. This allows the rest of the development team to keep working and the developer who wrote the code can fix it. Gated Check-In uses a hallmark TFS feature called the Shelf Set to accomplish this. When a Gated Check-In fails, the developer can ask other developers for help in fixing the code – all without the broken code ever being checked in.

## is verified by Testers…

Testers have traditionally not been part of the development team. Testing was an activity that happened after development was finished. The realization that testers are an integral part of the development team is just now dawning on organizations because of the very numbers mentioned at the beginning of this paper. Visual Studio 2010 fully incorporates testers into the heart of any project by including Test Case Management and a professional test management and execution tool – Microsoft Test Manager (MTM), shown in Figure 6.

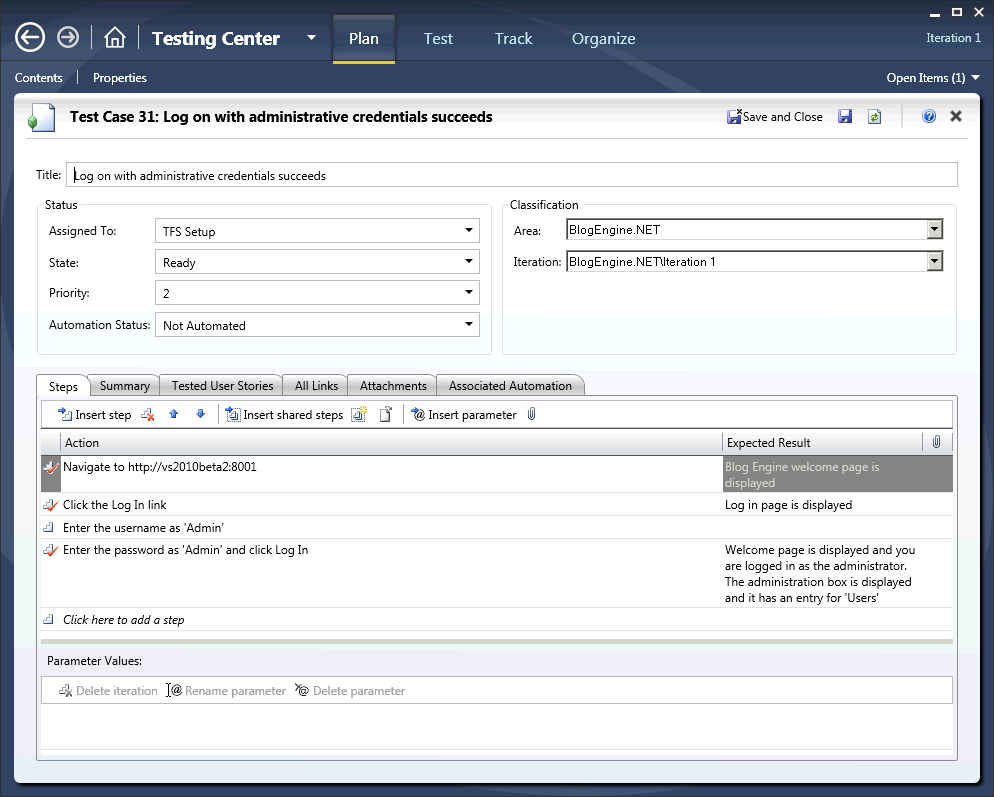


Figure : Microsoft Test Manager.

At the outset of a project, the QA team can associate test cases directly with requirements. This allows developers to see the test cases that will be executed. Later, testers can execute tests against the code to determine if it meets the customer needs. There are many types of testing that can help ensure the quality of your application. Manual or general testing makes up about 70 percent of all the testing carried out by organizations today but other test types are also available. These test types range from standard unit tests and performance tests to load tests and automated tests. As part of this, testers can collect detailed diagnostic information and videos of their testing session so that when a bug is filed it is actionable – that is, developers can actually fix the discovered defects. Figure 7 shows the test repro section of a bug filed through MTM. All of the steps are detailed including which steps passed, which failed, and any attachments or comments included with each step. The video time indexing allows developers to jump right to the spot in the video where the bug occurred. Also included is the IntelliTrace™ log file which provides detailed debugging information for the test session leading up to the bug.

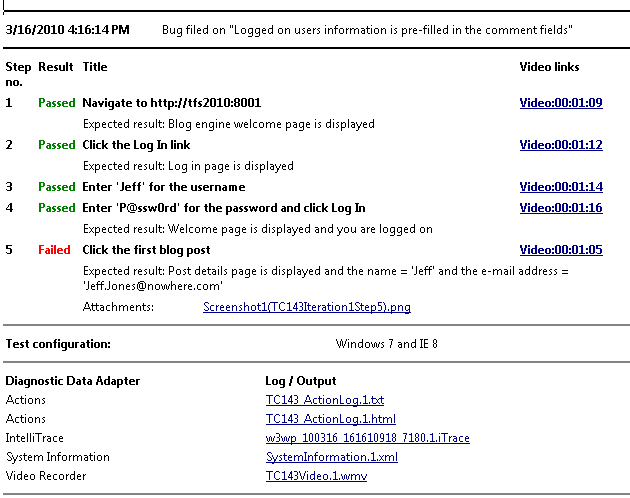


Figure 7 - Bug Repro Steps.

Test Impact Analysis is a new feature that provides testers with the information they need to select which tests are high priority tests based on the code changes that a developer makes. When code changes that has already passed testing, the testers are alerted and can re-execute those tests to reduce or eliminate regression bugs. Lab Management refines this further by virtualizing the testing infrastructure so that when a defect occurs a snapshot can be taken of the virtual environment and provided to the developer so he can easily re-create the bug in the environment in which it was found.

## … And is visible to Management

Visual Studio 2010 and TFS 2010 promote quality from the ground up – but management wants to know about the project status. Management has to report to customers on the progress and quality of the software and discovering at the last minute that there are problems can make it difficult to alter course. TFS includes detailed reporting based on SQL Server Analysis Services. When combined with Microsoft Office SharePoint Server and Excel Services, information – not just data – is always available, up-to-date and accurate. Figure 8 shows just one of the quality information dashboards available to you.

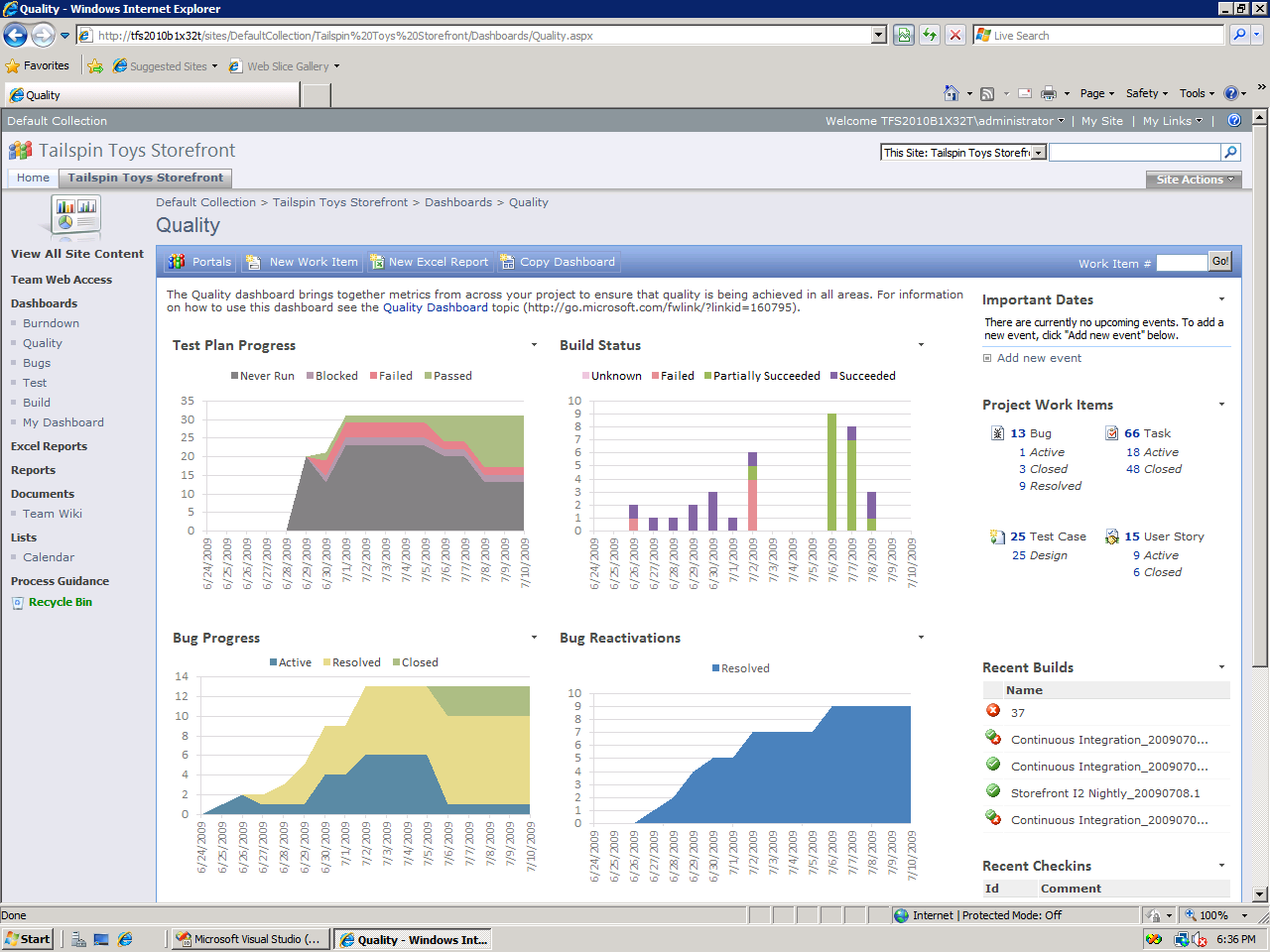


Figure 8 - Quality Metrics.

The Test Plan Progress shows how many tests there are, how many have been executed, and how many have passed. Build Status indicates the number of automated builds and their success over time – it combines build pass/fail with the results of automated testing that runs as part of the build. This provides a daily snapshot of the quality of your application. Bug Progress and Bug Reactivations provide detailed information on the status of all bugs filed against the application and how much re-work is being done because bugs are not being fixed correctly.

This information allows management to make course corrections early on. If a high number of bugs are being filed, actions can be taken to reduce the bug count. If tests are not being executed then more testers can be brought on. This provides the ability to better ensure a quality product is delivered on time and to the customers’ satisfaction.

# Quality Matters

Every development team approaches each project with one goal in mind – to produce a quality application. But that goal can be difficult to reach without the right supporting tools. From application architecture through development, test and maintenance, Visual Studio 2010 provides the right supporting tools to help ensure that applications are of the highest quality. No matter what role you play in the development process, these tools are built with you in mind to help you meet the needs of your customers.

# See what other are saying

**ICONICS**  
Global Software Developer Expects to Double Productivity with Development Solution

***ICONICS, a Microsoft Gold Certified Partner, provides industrial automation and visualization software to organizations in more than 60 countries. The company’s development teams include employees from locations in three countries—and the geographic distance created collaboration challenges. Developers lacked an integrated tool set that could facilitate project management, promote data access, accelerate development, and automate testing processes. In 2009, ICONICS enhanced its development environment by deploying Microsoft Visual Studio 2010 Ultimate, Visual Studio Team Foundation Server 2010, and Visual Studio Lab Management 2010. The fully integrated solution gives global teams unified tools to simplify project management and streamline product lifecycle processes. As a result, ICONICS can cut costs and improve product quality, and it expects to increase productivity by 100 percent.***

“The new testing capabilities in Visual Studio 2010 are revolutionary. Once we get more teams using the tools, I can see our productivity doubling. We can run automated tests overnight without anyone being present. In the future, we also expect that features like IntelliTrace will significantly reduce the time required for us to debug applications.” - Chris Elsbree, Chief Software Architect, ICONICS

“The advantages of using Visual Studio 2010 all come down to performance and quality. Features such as hierarchical work item tracking, branch visualization, and IntelliTrace—and our ability to build older versions with the new system—all contribute to an overall improvement in our efficiency. And this, ultimately results in a better quality product.” - Russ Agrusa, President and CEO, ICONICS

1. National Institute of Standards and Technology. (2002). *Planning Report 02-3, The Economic Impacts of Inadequate Infrastructure for Software Testing.* U.S. Department of Commerce. [↑](#footnote-ref-1)
2. Standish Group, (2009). *Chaos Report*. [↑](#footnote-ref-2)