

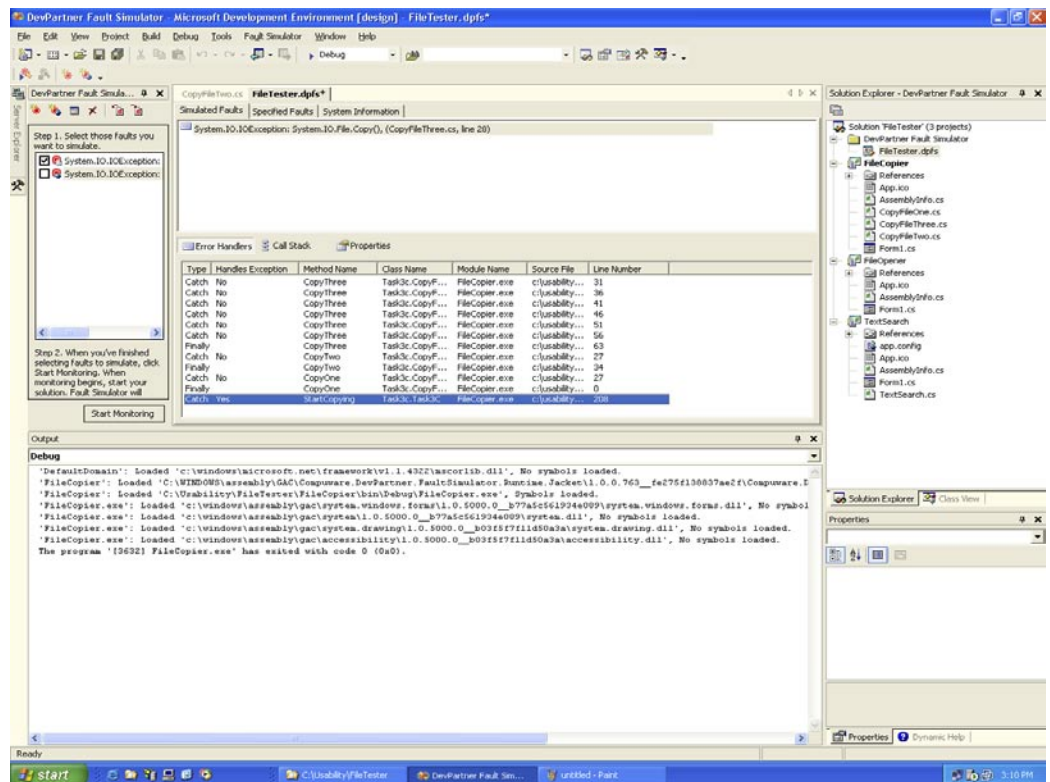
## A powerful new tool to help validate error-handling code

Compuware's new DevPartner Fault Simulator provides a new fix for the old problem of testing error handlers, creating a repeatable environment for proactively analyzing and debugging application error-handling code.

A significant and growing portion of an application's code is dedicated to error handling. Only a small portion of that error-handling code, however, is actually tested prior to deployment. Based on an industry standard of 10 to 20 bugs per each 1,000 lines of code, there is the very real possibility that a 50,000-line application could have as many as 150 to 300 undetected bugs hiding in its error-handling code.

Faulty error code handlers can cause unplanned application downtime, lost customer revenue, increased development costs and the risk of security vulnerabilities. Despite the many problems faulty error-code handlers can cause, testing them has lagged behind because quality assurance technicians have lacked an easy, repeatable and safe method to create faults within the testing and debugging environment.

The company that brought you DevPartner now has a solution to this long-standing problem. Compuware's DevPartner Fault Simulator is a unique developer tool using fault simulation to



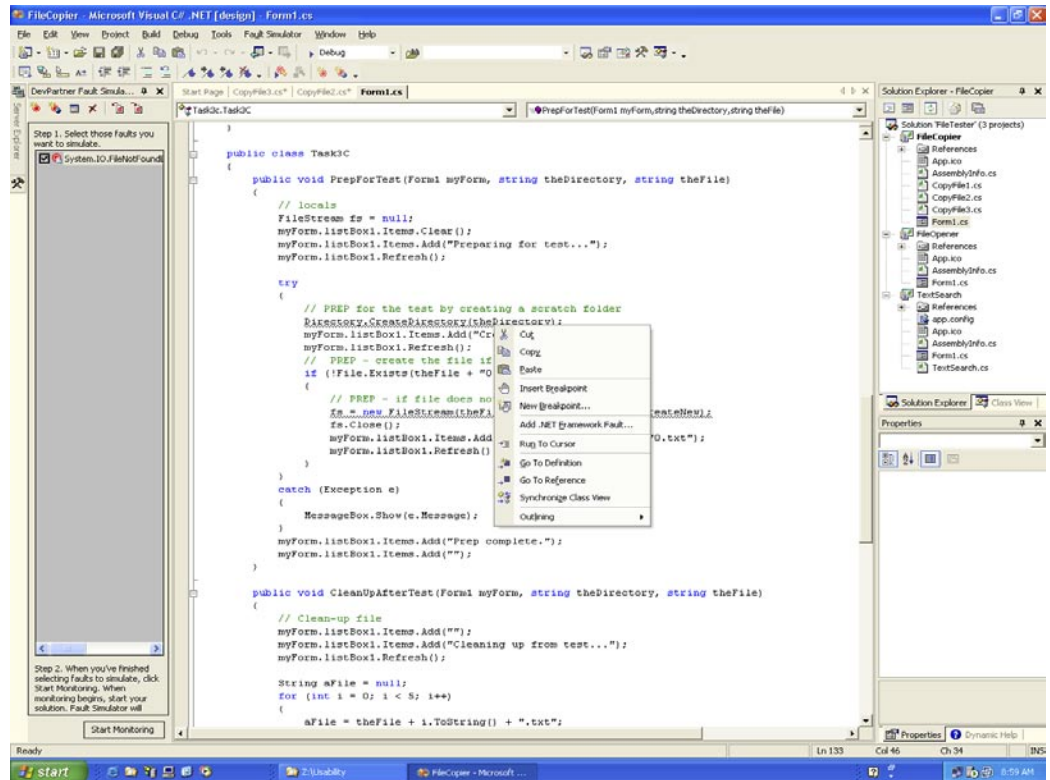
A simulation captures detailed information about the application's response for each fault simulated, including the error-handling stack.

emulate real-world application errors. DevPartner Fault Simulator allows developers to work in a predictable, repeatable environment to proactively analyze and debug application error-handling code—leading to a better end-user experience and eliminating the loss of revenue that comes with unplanned application downtime.

### Where errors originate, and why they're so difficult to identify

Errors that bring applications to a grinding halt can originate in multiple places. Some are the result of environmental problems, including file or database access, registry access failures, bad permissions, low

memory or disk space, or network issues. Others start in system and runtime libraries, where there are thousands of defined errors. Still more errors can be the result of the application itself, such as user-defined errors, bad return values, bad parameters, buffer overruns, out-of-range data, parameter mismatches and many more.



Source code is highlighted at the lines where faults can be simulated. A fault is added by simply choosing from the list provided.

The overwhelming number of errors that could threaten an application’s performance makes error-handling code a necessity. Yet developers have been forced to rely upon unpredictable and ad hoc methods for testing error handlers. This leaves developers facing several difficult issues when attempting to verify error handlers, including:

- identifying what errors can occur, along with where and when they occur
- attempting to trace error-handling execution
- creating repeatable tests
- testing error handlers manually is time-consuming
- tools are lacking for error simulation and analysis.

### How fault simulation works

Fault simulation is a safe, non-intrusive alternative to traditional testing methods. It is a technique used to validate the robustness of software application code by artificially inserting faults into the code, mimicking real-world failures. A simulation shows how error-handling code functions within the application. DevPartner Fault Simulation is also an excellent tool for regression testing, allowing developers to fix a bug once with the knowledge that it will stay fixed as the application evolves over time.

An example of the benefits of fault simulation is evaluating how an application handles a network failure. A developer could physically disconnect a critical cable from the network, or attempt to cause a failure by generating excessive network traffic. These actions can produce the necessary test conditions, but they are difficult to control precisely and are likely to create other problems in the process. The same outcome can be accomplished more safely by using DevPartner Fault Simulator to observe how the application would handle the failures generated by a simulated network failure. DevPartner Fault Simulator is designed to affect only a specific section of the code or application—not the entire operating system, as an actual network disconnect or overload might do.

### Providing insight for developers

DevPartner Fault Simulator is engineered to help developers do their jobs more efficiently in several ways, including user education, safe fault simulation and the creation of comprehensive results reports.

#### User education

DevPartner Fault Simulator highlights the areas in source code where faults can be simulated. Additionally, developers can select a line of code within Visual Studio .NET and allow DevPartner Fault Simulator to identify the list of exceptions that can be simulated at that location. This helps you to understand what faults your code must be built to handle.

### Safe fault simulation

A .NET Framework fault can be simulated either on a line of code or one that is independent of location using DevPartner Fault Simulator. Developers can also simulate an environmental failure in the target application. Properties, parameters and conditions associated with every fault make it possible to further refine the simulation.

### Comprehensive reports

DevPartner Fault Simulator displays information about the faults being simulated and how they are being handled as the simulation is taking place. It also provides comprehensive results about the error handling in the application code upon completion of the simulation. The results can be accessed from within the debugger or operating environment. This includes a description of the fault, the current call stack and an error-handler stack for each instance simulated.

### Three ways to use the tool

DevPartner Fault Simulator is available for use in three distinct ways by developers who are running DevPartner Studio:

- integrated into Visual Studio .NET—to analyze and debug error handlers in source code
- run as a standalone—to simulate faults in a running program
- executed from the command line—using scripts and batch files to automate the testing of applications.

### Integrated into Visual Studio .NET

DevPartner Fault Simulator helps developers test and debug error handlers when it is integrated into Visual Studio .NET. Importantly, DevPartner Fault Simulator is able to simulate faults without disrupting the debugger, operating system or the .NET development environment. It records how the program reacts to an error, allowing users to trace program activity from where the error occurs to where it is ultimately handled. The information is displayed as the simulation proceeds, with the final results subsequently stored in a results file—which is available for immediate viewing and is also saved for later review. In some cases, DevPartner Fault Simulator also allows developers the chance to go to the original source statement to better debug and fix fault-handling problems.

### Run as a standalone in QA

DevPartner Fault Simulator is available as a standalone to help quality assurance technicians and developers simulate faults in a target application. It can be used to choose environmental faults from a user-selectable list, and also to specify properties that will trigger the selected fault. The resulting report can be provided to development to resolve any defects.

### Executed from the command line

DevPartner Fault Simulator makes it possible to automate fault simulations on projects that do not require user intervention. You can execute scripts to automate test applications—such as part of an overnight build or test automation system—right from the command line. This supports any fault sets that have been previously configured in DevPartner Fault Simulator.

DevPartner Fault Simulator can be used to run an interactive fault simulation, or it can be set up to execute from the command line with Compuware TestPartner—an automated testing tool that accelerates the functional testing of business-critical applications.



Environmental faults may be simulated outside the IDE enabling access to the simulator by other organizations.

