TMurgent Technologies
26 Angela Street
Canton, MA 02021  USA
(+1) 781.492.0403

# When Applications Crash Part II - WER

A two part guide to how any ISV can find and fix crashes that occur at customer sites.

Tim Mangan
Founder, *TMurgent Technologies*
April 1, 2006

## Introduction

Unfortunately it is inevitable. No matter how careful your developers are. No matter how thorough your testing is. It will happen. At a customer site. Before they buy your software.

Something crashes in your software. Buffer overrun. Dereferencing a null pointer. Divide by zero. You name it, it can happen – and will at some point.

This white paper, and its companion[1], covers the basics of two methods you can use to solve these problems when they happen. Both require some advance planning on your part.

The first method, covered in Part I of this paper, is an old tried-and-true method of using DrWtsn32. This second method, covered here in Part II, is newer and usually easier for your customer to deal with.

## Method 2 – Windows Error Reporting and WinQual.

Getting Watson dumps from a customer can be painful. Fortunately, Microsoft has introduced the Windows Error Reporting Service (WER) in Windows XP, Server 2003, and subsequent versions of the OS. WER is not available on Windows 2000 professional or server, so you will need to continue to use Watson dump for those customers.

Windows Error Reporting (WER) is that annoying popup dialog that appears when an application crashes that usually says "…has encountered a problem and needs to close". It then asks if you want to send an error report to Microsoft. If the application is a background service, error reporting hangs onto the exception information and notifies the next administrator to log into the system. The Microsoft Knowledge Base Article on enabling and disabling Windows Error Reporting is KB310414[2]

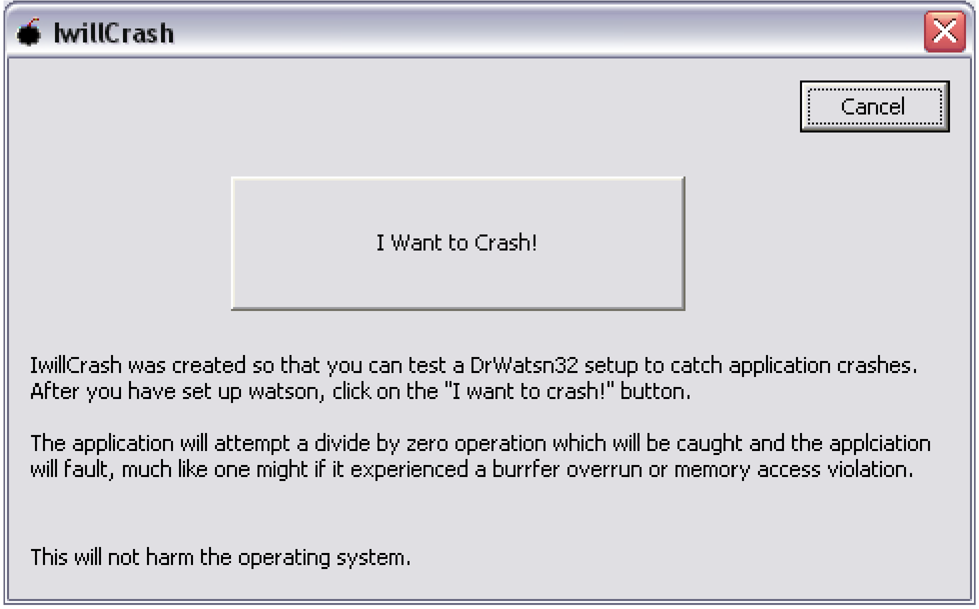
If you thought that the Windows Error Reporting service was only for Microsoft bugs, then you are in for a surprise. Microsoft has a program, albeit rather stealth, to work with ISVs. It is part of the Microsoft "WinQual" program.

### Causing a WER

I wrote a test program that will crash on demand. (This was actually harder than I thought it would be because the code optimization kept optimizing out the divide by zero

[1] See *"When Applications Crash, Part II" www.tmurgent.com/WhitePapers/WhenAppicationsCrashP1.pdf April 2006.*
[2] http://support.microsoft.com/kb/310414/EN-US/

on me!) The program simply displays the dialog shown in Figure 1 and will crash if you click on the big button.



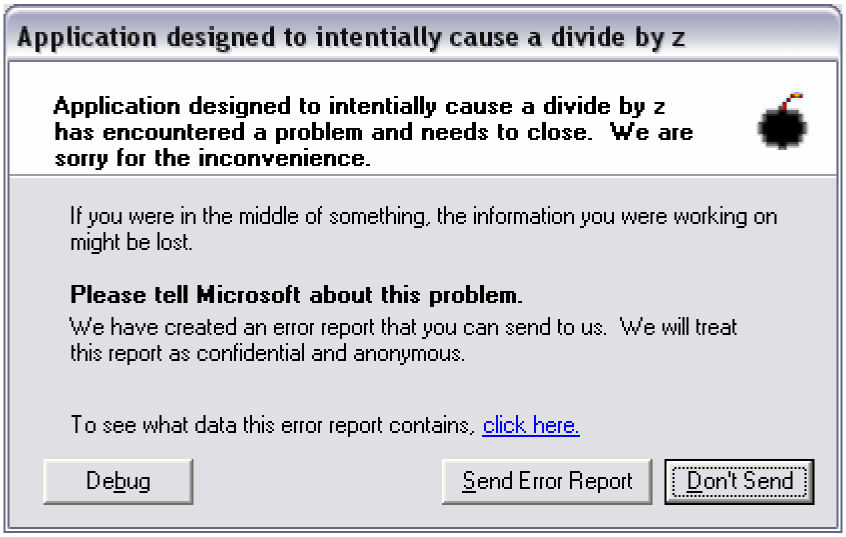
IwillCrash was created so that you can test a DrWatsn32 setup to catch application crashes. After you have set up watson, click on the "I want to crash!" button.

The application will attempt a divide by zero operation which will be caught and the applciation will fault, much like one might if it experienced a burrfer overrun or memory access violation.

This will not harm the operating system.

**Figure 1 - test crash program**

If you click on the button on Windows XP or 2003, the program crashes Dwwin runs and a dialog similar to the one shown in Figure 2 appears:



**Application designed to intentially cause a divide by z has encountered a problem and needs to close. We are sorry for the inconvenience.**

If you were in the middle of something, the information you were working on might be lost.

**Please tell Microsoft about this problem.**
We have created an error report that you can send to us. We will treat this report as confidential and anonymous.

To see what data this error report contains, click here.

**Figure 2 - WER dialog**

I should mention that some things, like windows services that run in the background, might not report with this dialog right away. Typically, Windows will retain the information and report the error the next time any administrator logs into the system. Also, on a production system without a secondary debugger installed (this screen capture was from a development system) the Debug button would not appear.

Dwwin displays this dialog and gives the user an opportunity to see what information is collected and included in the report. For all practical purposes, this information is the same as a Watson mini-dump. The "click here" link on the dialog allows the user to see what information is to be sent, and has an additional link that takes users to a Microsoft page (http://oca.microsoft.com/en/dcp20.asp ) describing the policies associated with the collected information.

If the user chooses to send the information to Microsoft, it is collected and analyzed. If an ISV registers for the Winqual program (which is free, at present) the ISV can access these reports.

## Joining Winqual

Information about this program is available from the following link (currently) https://winqual.microsoft.com/default.aspx . Once an ISV qualifies for the WinQual program, it is possible to retrieve dmp files from the WinQual website.

There is a catch. Bugs are like underwear, you don't show them in public. You need to verify your right to get these caught bugs. This means you must sign your executables with a digital signature from a root CA that Microsoft supports. In fact, you need a supported digital id just to get registered into the WinQual program. WinQual is free, but the ID isn't. Currently Microsoft only supports Verisign authenticode IDs, which are about $500 per year. Microsoft has indicated that they plan to expand to support other root certificate authorities, but as of this writing you must use Verisign. (Check the winqual site for updates on that).

By the way, if you write any device drivers, you will need one of these to support x64 vista drivers anyway as that OS will no longer load unsigned drivers when the product is released.

In the end, you still need to keep the source and symbols around. But now you can tell your customer to send the report to Microsoft the next time it happens and you can get it from there. Not bad!

Winqual is an umbrella program that supports more than just WER. Hardware, Software, and Driver developer organizations can use Winqual for not only WER, but for Logo programs ("*designed for…"*) and market listings. See the Winqual site (https://winqual.microsoft.com ) for current information on the Winqual program.

## Preparing your executables

Developers should take a look at the "Windows Error Reporting for Developers" (https://winqual.microsoft.com/help/wer_help/dev.aspx ) for information on preparing there executables prior to shipping.

ISV software crashes are not automatically saved and categorized by Winqual. You have to register a program or Microsoft with through the crash away. There are basically two ways to do this.

Digitally signed code that has been accepted via a windows logo program is automatically registered and will be retained.

Other software can be registered by name and version on the Winqual site. It is not (currently) a requirement to digitally sign those executables, but it probably is a good idea anyway (alter all, you had to pay good money for your company id to get into the program). This registration is called "File Mapping". When you request a file mapping the web site mentions that it takes about 5-7 days to get the file mapping approved. Until approved, dumps will not be saved.

The developer information linked to above also claims that you need to add the ReportFault API (faultrep.dll) at the top level of your program to install an exception handler in order to use WER. This is only needed if your application already has exception handling that interferes with WER. In other words, if you perform exception handing in your code and continue to run rather then crash, there won't be a WER report. By using this API, you can have your exception handler send in a fault report with a dump at the time of the exception, and still continue to run. I have seen this work in the Office 12 beta, and the user might not even be told about the problem until later on. In some cases, you also might be able to enhance the information captured by doing some simple development in your program and the ReportFault API also A cheesy method would be to place string information onto the stack that will be easily visible in the debugger. You need to be careful, however – you already faulted once! . Note that using the ReportFault API is totally optional and is not necessary to obtain the crash reports when your application crashes. Although it is dated, there are slides and a transcript from a Microsoft Webcast (PSS ID Number: 813510) that talks about all this.

## Mapping Files (Registering for Dumps)

The Windows Error Reporting Service collects crash dumps sent in. Microsoft runs a quick automated look at the dump to categorize it. If it knows about this problem and there is a response registered with it (see "Setting Up User-responses" below) the user can receive information about the specific problem.

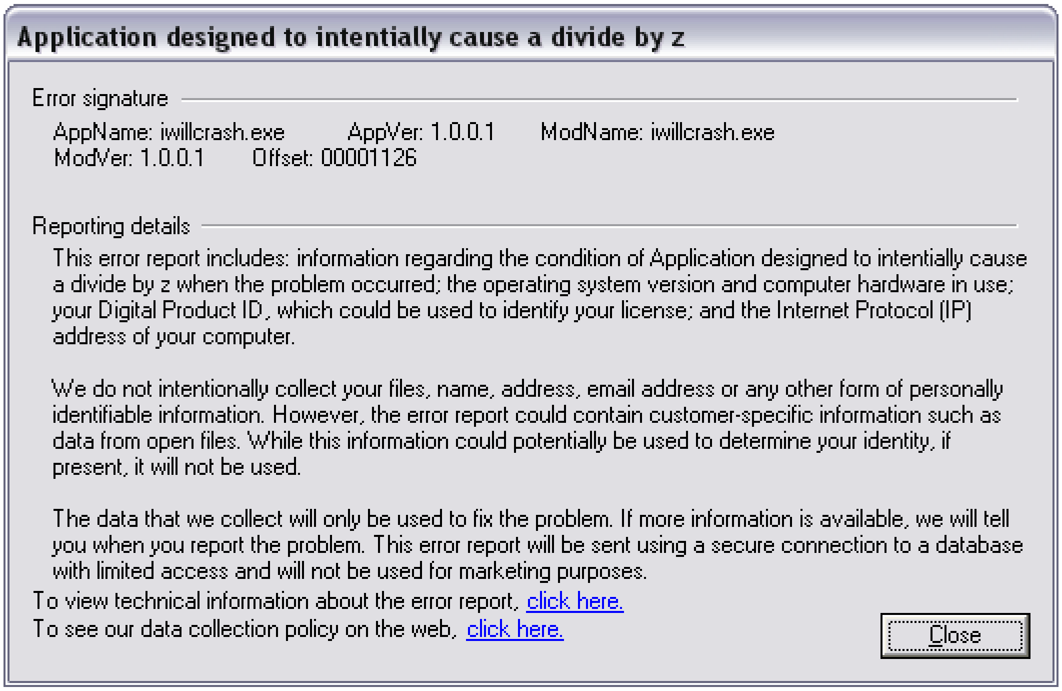
When the application is fully categorized, information is databased so that you can query crashes based on a variety of criteria. Microsoft has thought this through to support large, consumer oriented ISVs that have lots of products, version, and problems. Hopefully you and I don't need all this fancy stuff.

It can take a day for crashes to be fully categorized. You also will not see information regarding a single crash. WER currently only shows issues for which there are three or more crashes seen. This is down from 50, so it might change again in the future.

Crashes are categorized based on an “error signature””. This signature is defined using five parameters:

- Application Name
- Application version
- Module Name
- Module Version
- Offset into module

For the example crash in figures 1 and 2, we can see the signature can be viewed by pressing the “click here” link in the dwwin dialog:



Application designed to intentially cause a divide by z

Error signature

AppName: iwillcrash.exe    AppVer: 1.0.0.1    ModName: iwillcrash.exe
ModVer: 1.0.0.1    Offset: 00001126

Reporting details

This error report includes: information regarding the condition of Application designed to intentially cause a divide by z when the problem occurred; the operating system version and computer hardware in use; your Digital Product ID, which could be used to identify your license; and the Internet Protocol (IP) address of your computer.

We do not intentionally collect your files, name, address, email address or any other form of personally identifiable information. However, the error report could contain customer-specific information such as data from open files. While this information could potentially be used to determine your identity, if present, it will not be used.

The data that we collect will only be used to fix the problem. If more information is available, we will tell you when you report the problem. This error report will be sent using a secure connection to a database with limited access and will not be used for marketing purposes.

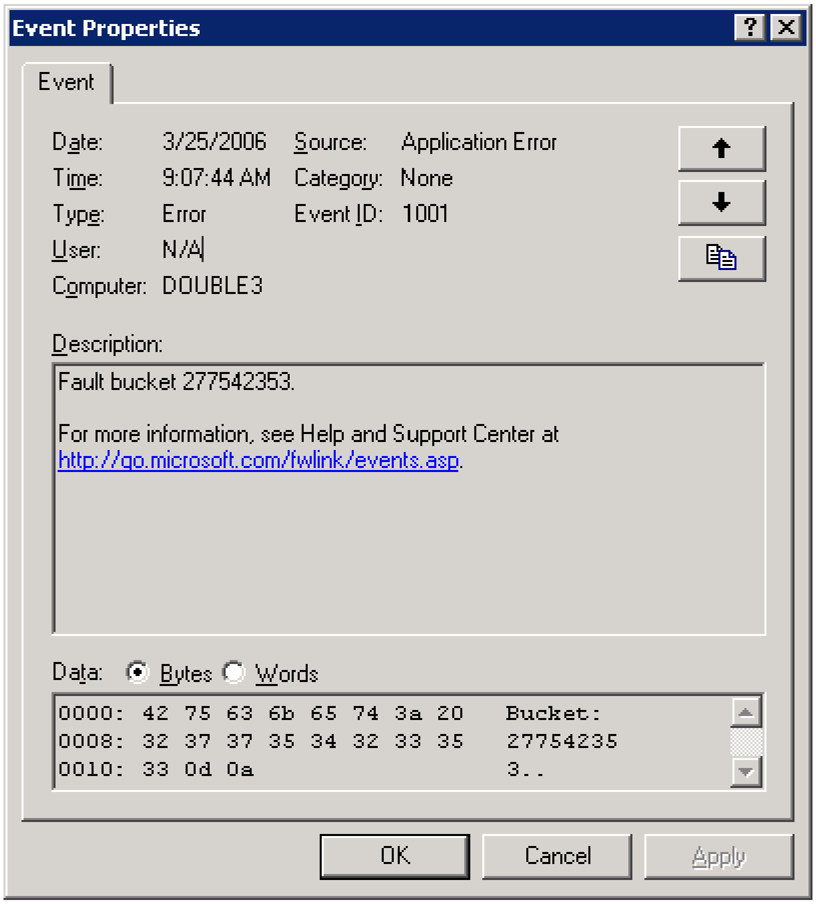
To view technical information about the error report, click here.

To see our data collection policy on the web, click here.

**Figure 3 - WER "click here" information**

If the crash had happened in a dll loaded by the application, we would have seen the name and version of that dll in the ModName and ModVer fields.

When a user chooses to send in a report it is immediately bucketed, and an event is added to the users Windows Event Messages – Application Category. An example of such a message is shown in Figure 4. If you have trouble getting your dumps out of winqual obtaining this number from the user may help troubleshoot that.



Date: 3/25/2006 Source: Application Error
Time: 9:07:44 AM Category: None
Type: Error Event ID: 1001
User: N/A
Computer: DOUBLE3

Description:

Fault bucket 277542353.

For more information, see Help and Support Center at http://go.microsoft.com/fwlink/events.asp.

Data: ● Bytes ○ Words

```
0000: 42 75 63 6b 65 74 3a 20   Bucket:
0008: 32 37 37 35 34 32 33 35   27754235
0010: 33 0d 0a                  3..
```

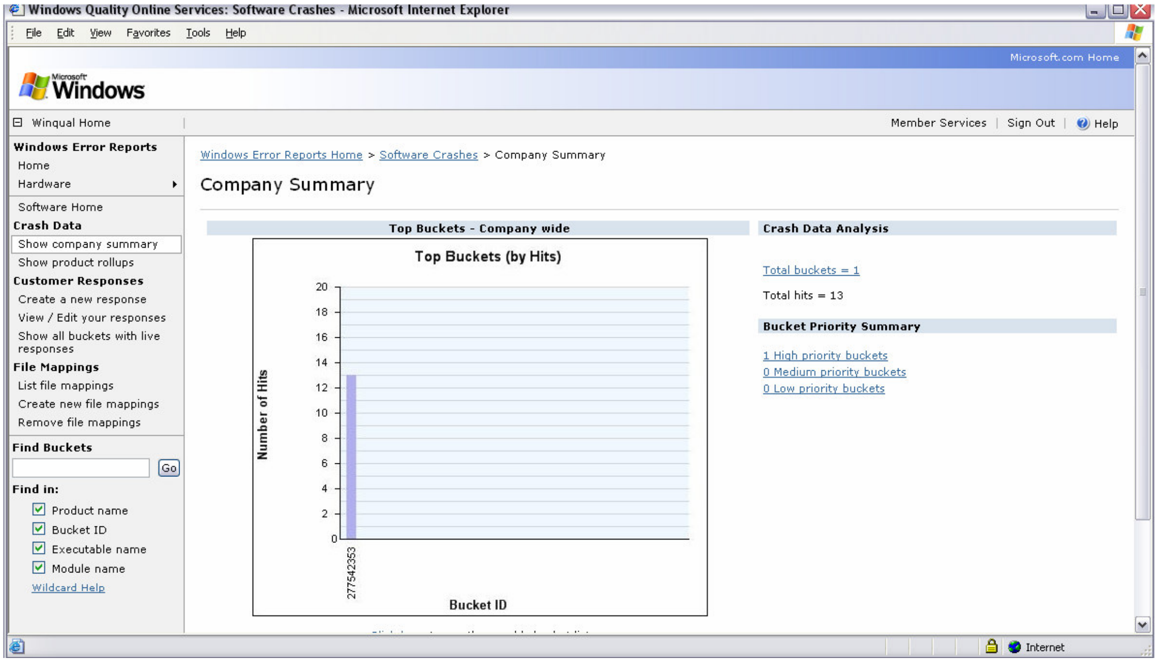
**Figure 4 - Error bucket stored on user system.**

## Accessing Dumps

Once the crash is transmitted to Microsoft, the categorization process will place the dump in one or more “buckets” based on the signature parameters. . Again, it may take a day after the user submits a report before an ISV can see that that crash in the reports.

Error reporting to the ISV is based on these buckets, and allows you to query reports on their basis. The contents of information dump is contained in a CAB file. My sources inside Microsoft indicate that the program is about to be updated, so I won’t go into the process of obtaining the CAB file – see the Winqual site for the latest details there.

When everything is in place and crashes occur, you can access them via the Winqual website. Figure 5 shows the company summary interface where you can view the most frequent problems.



Crash Data Analysis

Total buckets = 1

Total hits = 13

Bucket Priority Summary

1 High priority buckets
0 Medium priority buckets
0 Low priority buckets

**Figure 5 - Winqual report showing results**

At this point Microsoft is only categorizing the issues with the mapped executable. It is not saving the cab files once analysis is complete. You must click on a bucket and then request cab download. Only then does Microsoft flag the bucket to retain crashes. The next crash sent will be retained. Microsoft has a policy on the number that will be retained and how long they will be retained, however that policy will probably change shortly.

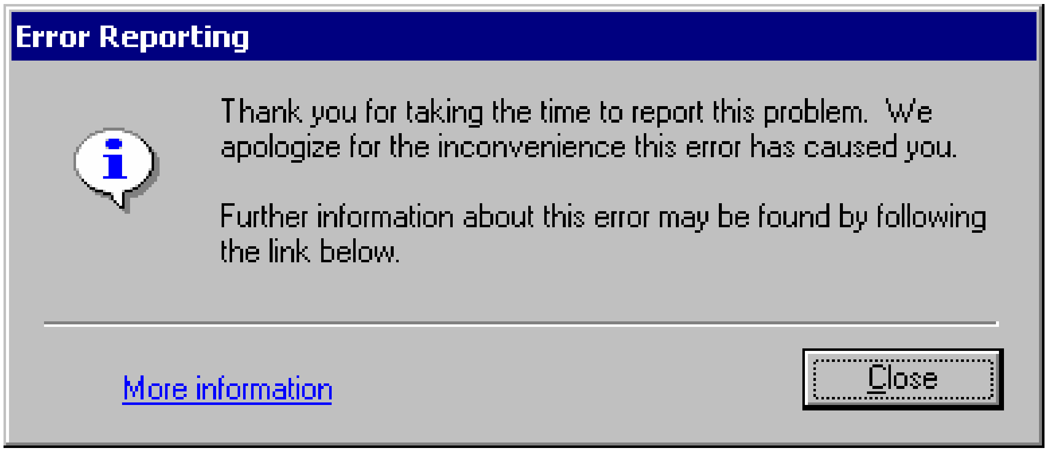
NOTE: Currently (March 2006), you have to be fairly patient with Winqual. Once you are in the program, getting a mapping registered will take several days to a week. After that, even though the Winqual folks acknowledge your mapping and the Winqual web-site reflects it, it might take still another week before it is implemented in the Winqual infrastructure to be able to hang on to your crashes. Even then, it takes some time (at least 1 day) before captured crashes are available to you from the website.

My sources within Microsoft indicate that a new refresh of the Winqual system is being worked on that should reduce all delays to 24 hours.

## Setting Up User-responses

Probably the best part of WER is the ability to set up responses. If you have a problem with an application in the field and you diagnose it, you can get a message out to users who have the same problem in the future!

The ISV can register a message associated with a particular signature. When additional users report a matching problem, after sending the file to the WER service it will run a very quick automated check to see if a matching issue response exists. If so, that information will be conveyed back to the user via dwwin. The dialog box below shows an example of the message received by the user when there is a response. By clicking on the "More information" link, you can provide an HTML message.


Error Reporting

Thank you for taking the time to report this problem. We apologize for the inconvenience this error has caused you.

Further information about this error may be found by following the link below.

**Figure 6 - WER User Dialog when response is available**

The advice in this response page should be something like a work-around or requesting that they upgrade to a fixed version. You can even include a link to the download if you host it on your web servers. You can also include a redirection back to your own site to a form input page so that you can ask the user for additional information that might be helpful in diagnosing the problem.

Also, if you are a Winqual member, updated device drivers can also be distributed via Microsoft's Driver Distribution Center for Windows Update.

## *Summary*

This white paper, in two parts, describes the basic techniques available to obtain information to enable the company to debug field issues that occur with their software products.

Obviously, these techniques are not a substitute for a good quality system for producing software. ISVs need to continue to invest time, energy, and money into quality programs and procedures. Experience has consistently shown that it is cheaper to invest in preventing problems from occurring in the field than to fix them after the fact.

But even the most thorough quality systems fail to catch every problem. Being prepared to address issues that occur in the field is an important part of that quality program.