



# Testcover.com

## Dedicated to Excellence in Test Design

**Testing Breakthrough Needed.** Every day we become more dependent on software, networks and distributed applications. In just a few years the Internet has evolved from research tool to business necessity. Our shopping, banking, appliances, and health care are increasingly automated by systems that *must* work correctly. It is an unfortunate fact that along with this wonderful progress there are software problems causing lost productivity, power outages, recalled cars, even personal injury, because systems do *not* always work correctly.

Our test organizations and their tools and processes have not kept up with the rapid proliferation of new technologies and applications. Our expectations for more features in shorter development cycles have challenged our ability to manage the complexity of the work. There is no single cure-all for this situation. Some have observed that we need advances in testing as we have seen in computing hardware, networks and software development in recent years.

**The Testcover.com service is a truly innovative advance for test coverage and efficiency.**

**Testcover.com Advances Test Design.** The Testcover.com service is a truly innovative advance for test coverage and efficiency. It uses a *covering* test case generator, which means that its test designs are statistically complete. All pairs of test values are guaranteed to be included in the test cases.

You don't worry about whether this display resolution is tested with that operating system version, because all the pairs of test values are included. The test design's ability to find defects is *improved* compared to others depending on random selection or guesswork.

You might think that this improved coverage would require more test cases and thus more time and work. But the opposite is true! The test case generator works to minimize the number of test cases. Since the test design is complete with a minimal number of cases, you know when there are "enough" test cases. Testers can spend less time on this part of test planning and end up running fewer test cases. Improved test efficiency is the result.

**So What's New?** Experienced software engineers know people have been struggling with this test design problem for many years. One of the first was Dr. Genichi Taguchi, who introduced the use of orthogonal arrays to select test parameters. Since then several tools have been offered to assist in test design. Although there have been advances, several drawbacks have prevented widespread adoption of these tools. There are four main problem areas.

**1. Finding an Array.** The tool must find a *covering array* to use as a template for the test case list. Generating covering arrays can be a computational challenge: Finding the best arrays continues to be an area of active research. Practical test case generators must have sufficient range and speed to handle everyday test jobs quickly and efficiently. The Testcover.com service uses recent research advances<sup>1</sup> as well as proprietary technologies to generate test cases.

**2. Handling Constraints.** What happens if the test plan calls for running the Internet Explorer browser on the Linux operating system? The configuration is not supported; any test case including the pair is impossible. But you cannot just skip such bad test cases because they contain other pairs of test values which are required. Practical test case generators must have a simple way to cover all *allowed* pairs while avoiding impossible combinations. Testcover.com handles constraints with a simple input form that describes the allowed combinations of test values.

<sup>1</sup> *New Vector Operator Methods for Covering Array Construction*, George B. Sherwood, submitted for publication.

**3. Supporting Error Cases.** The third challenge for test case generators is that systems under test have multiple operational states. A calendar application behaves differently with a good date (e.g. Jan 31, 2003) than with a bad date (Jan 32, 2003). If the combination "Jan with 2003" is tested only in the error handling state, there is a gap in the test coverage of the normal state. Practical test case generators must allow for different operational states to cover the system under test completely.<sup>2</sup> Each Testcover.com service request supports multiple *partitions* of test cases for the multiple operational states.

**Testcover.com offers a covering test case generator that's practical for everyday use.**

**4. Operating Simply.** Working at the end of the development project, often with compressed schedules, testers must be as efficient as possible in discovering bugs and in proving the product will work as designed. There is no time for a tool that requires a great deal of set-up or a significant effort to run. The Testcover.com service provides a simple yet versatile interface, enabling you to handle error cases and constraints among test values easily. Testcover.com offers a covering test case generator that's practical for everyday use.

**An Invitation.** Currently Testcover.com is offering a Risk-Free Trial of our exclusive test case generator service. New subscribers can see for themselves how the service can reduce test costs and improve product quality. We're confident that if you try the service in your testing situation, you will see how it can improve your test process and provide a needed advance in the critical job of testing.

<sup>2</sup> Finite state models have been used successfully to specify tests for communications protocols. However many real-world systems do not enjoy the complete specification needed for such an analysis.

**An Example.** This calendar application example illustrates the use of the Testcover.com service with constraints and with error conditions. There are two operational states to be tested:

1. Good dates, and
2. Error dates with months that are too long.

In the request below, the lines following those with the pound character (#) describe the test values for the two different operational states. In each of these partitions, blocks of three lines list allowed values for each of the three test factors (Month, Day and Year). Any combination of test values in a block is allowed.

**Calendar Example Request.**

```
Calendar Example
Month
Day
Year
#ok All good dates
jan feb mar apr may jun jul aug sep oct nov dec
1 10
2003 2004 2005
+ long month last day
jan mar may jul aug oct dec
31
2003 2004 2005
+ short month last day
apr jun sep nov
30
2003 2004 2005
+ feb last day
feb
28
2003 2005
+ leap day
feb
29
2004
#err Too long month
+ too long long month
jan mar may jul aug oct dec
32
2003
+ too long short month
apr jun sep nov
31
2004
+ feb too long, regular year
feb
29
2005
+ feb too long, leap year
feb
30
2004
```

Resulting test cases for the good dates are as follows.

**#1. All good dates**

Test	Case ID	Month	Day	Year	Combo Countdown
	ok1	jan	1	2003	123
	ok2	jan	10	2004	120
	ok3	feb	10	2003	117
	ok4	feb	1	2005	114
	ok5	mar	1	2004	111
	ok6	mar	10	2005	108
	ok7	may	31	2005	105
	ok8	aug	31	2004	102
	ok9	oct	31	2003	99
	ok10	nov	30	2004	96
	ok11	apr	30	2005	93
	ok12	jun	30	2003	90
	ok13	feb	29	2004	87
	ok14	apr	1	2003	85
	ok15	apr	10	2004	83
	ok16	may	10	2003	81
	ok17	jun	1	2004	79
	ok18	jun	10	2005	77
	ok19	jul	1	2003	75
	ok20	iul	10	2004	73
	ok21	aug	10	2003	71
	ok22	aug	1	2005	69
	ok23	sep	10	2003	67
	ok24	sep	1	2004	65
	ok25	oct	10	2004	63
	ok26	nov	10	2003	61
	ok27	nov	1	2005	59
	ok28	dec	10	2003	57
	ok29	dec	1	2004	55
	ok30	jul	31	2005	53
	ok31	dec	31	2005	51
	ok32	jan	31	2005	49
	ok33	mar	31	2003	47
	ok34	sep	30	2005	45
	ok35	feb	28	2003	43
	ok36	may	10	2004	42
	ok37	may	1	2005	41
	ok38	oct	1	2003	40
	ok39	oct	10	2005	39
	ok40	feb	28	2005	38

Error test cases for months that are too long are listed below.

**#2. Too long month**

Test	Case ID	Month	Day	Year	Combo Countdown
	err1	jan	32	2003	96
	err2	apr	31	2004	93
	err3	feb	29	2005	87
	err4	feb	30	2004	84
	err5	mar	32	2003	82
	err6	may	32	2003	80
	err7	jul	32	2003	78
	err8	aug	32	2003	76
	err9	oct	32	2003	74
	err10	dec	32	2003	72
	err11	jun	31	2004	70
	err12	sep	31	2004	68
	err13	nov	31	2004	66

**About Testcover.com.** Testcover.com, LLC was founded in 2003 to aid testers in the selection of efficient sets of test cases. Our staff has over 25 years of engineering experience on commercial development projects, and is responsible for the conception and development of CATS<sup>3</sup>, one of the first covering test case generators. Recent work also has led to significant improvements in the construction algorithms for covering arrays<sup>1</sup>, which are the mathematical foundation for test case generation.

You can find us at these addresses.

<http://www.testcover.com>

Testcover.com  
 41 Clover Hill Road  
 Colts Neck, NJ 07722-1007  
 USA

<sup>3</sup> *Effective Testing of Factor Combinations*, George Sherwood, Third International Conference on Software Testing, Analysis & Review, May 8-12, 1994, Washington, DC.