

Reducing Time to Market with Combinatorial Design Method Testing

Jerry Huller
Raytheon Company
16800 East CentreTech Parkway
DN, Bldg. 485, M/S 6109
Aurora, Colorado 80011-9046 USA
cjhuller@west.raytheon.com, j.huller@ieee.org

Abstract. Testers face the challenge of doing as much testing as possible within the available, and usually constrained, schedule. In the hyper-competitive commercial marketplace, it is not practical to exhaustively test all combinations of system test cases. This paper discusses the author's experience with a method that generates a small subset of test cases which provides good coverage of the test domain—the Combinatorial Design Method. It has proven to be flexible for system level testing of small, commercial satellite ground systems. Cost savings of 67% and schedule savings of 68% are shown for an example.

INTRODUCTION

This paper will briefly describe the business market and the configuration of a typical ground system, also known as a ground station. The elements of redundancy will be identified. The need for testing of the redundant combinations leads to a discussion of the Combinatorial Design Method technique for generating test cases. A general example and a specific example for a recent ground system are given. Cost and schedule savings for testing are illustrated. Manual and automatic methods of generating test cases with the Combinatorial Design Method are identified. Some questions for further investigation are posed. Management implications are described.

BUSINESS MARKET

Despite some setbacks in 1999 for the mobile telephone satellite industry, the satellite business should continue to boom. C.E. Unterberg, Towbin projects that the commercial satellite industry will grow at 17.9% annually, to \$114.9 billion in 2003 (PRNewswire 1999). The Teal Group predicts that almost 1,500 commercial communications satellites valued at \$126.8 billion will be launched from 2000 through 2009 (Aerospace Daily 1999).

These new satellites will require new ground systems or upgrades to existing ones. Raytheon Company is a major, global provider of hardware and software solutions for commercial customers who

operate satellite ground systems. Historically, these multi-million dollar projects have been characterized by short duration (approximately twelve to eighteen months, depending on the launch date for the satellite), reuse of code and previous system architectural designs, redundancy, tailoring of the design to satisfy international customer needs and constraints, and a small, core team (7-16 people).

A GROUND SYSTEM

An example ground system consists of three sites, as depicted in Figure 1. One site is the primary Satellite Control Center (SCC). Operators in the SCC control and monitor one or more orbiting satellites. A backup SCC at another site provides redundancy for satellite control. Co-located at the backup SCC is the primary Telemetry, Command, and Ranging (TCR) site. The TCR site contains the hardware, including antennas, for communicating with the satellites. A backup TCR site provides redundancy.

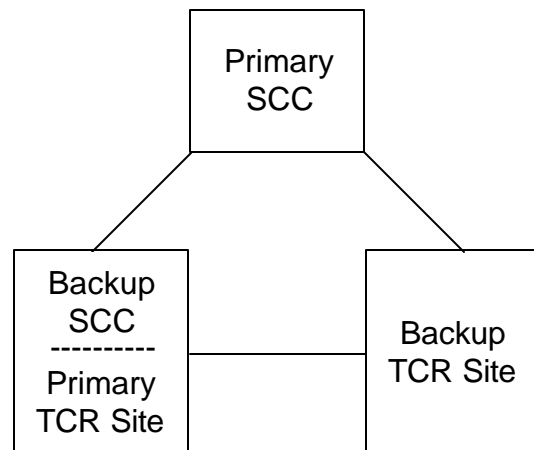


Figure 1. Example Ground System

A simplified block diagram of a ground system is shown in Figure 2. The Computer subsystem consists of the file servers, workstations, and local area network that run the application programs for controlling and monitoring the satellite and associated ground equipment. The Baseband subsystem consists of baseband units (BBUs) and

time code devices, such as Global Positioning System (GPS) units, for time stamping the telemetry data. The BBU is used to generate commands or ranging signals which are sent to a satellite, and to process telemetry or ranging signals which are received from a satellite. The term *baseband* refers to the frequencies used by a BBU. The Radio Frequency (RF) subsystem consists of hardware to uplink and downlink signals to and from a satellite via the Antenna subsystem. Typical antenna sizes for commercial communications satellites are 9–13 meters in diameter. The Site Communications Network subsystem contains the hardware and communications links, such as leased telephone lines, to interconnect the SCC and TCR sites.

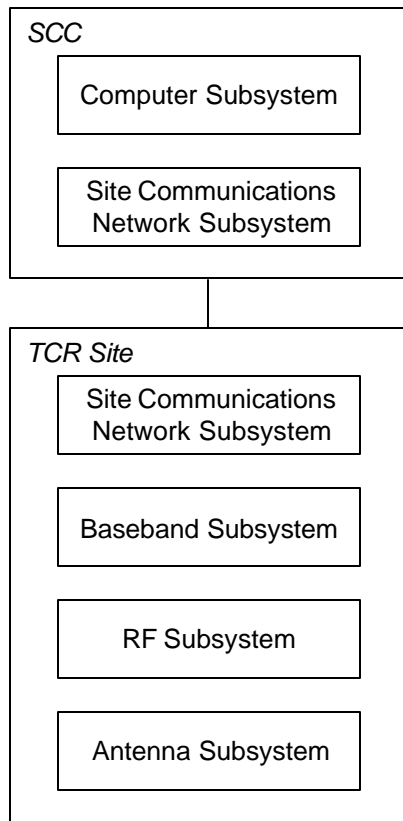


Figure 2. Ground System Subsystems

The application software in an SCC provides such capabilities as the following: commanding a satellite, processing telemetry, determining the range to a satellite, status and controlling the ground hardware to support satellite operations, monitoring the status of satellite subsystems, performing maneuvers to keep the satellite in its designated orbital location (referred to as stationkeeping), and operator training via a simulated satellite. The application software generally resides on a file server and is accessed via client workstations in the SCC.

To meet availability requirements for a ground system, redundant equipment is provided. As described earlier, the SCC and TCR sites have backup locations. The Site Communications Network

subsystem provides multiple communications paths for the data between an SCC and its TCR site. For example, as shown in Figure 1, data from the primary TCR site can be routed to the primary SCC via the backup TCR site if the communications link between the primary SCC and primary TCR site becomes nonoperational.

Within the primary SCC, there is typically a primary server and a backup server. At the backup SCC, there is a remote server. The same software is installed on all servers. Key operational data is backed up from the primary server to the other servers, so that any server can assume operational control within a short time period.

All workstations in an SCC are redundant. A workstation can run any of the application programs.

The primary and backup TCR sites often contain identical equipment for the Baseband, RF, and Antenna subsystems. However, the antennas may be different sizes or models within the Antenna subsystem. Within each Baseband subsystem, a spare time code unit and a spare BBU are usually provided. For an SCC controlling two satellites, the Baseband subsystem may include three BBUs. One BBU is dedicated to each satellite, with the third serving as a redundant backup for either satellite. The RF subsystem contains redundant hardware for uplinking and downlinking signals to and from the satellite via the online antenna.

Redundancy is also provided in communications between the ground system and the satellite. Two telemetry streams at different frequencies are typically provided.

To summarize, redundancy is provided in the following:

- SCC sites
- TCR sites
- File servers at the SCCs
- Workstations at the SCCs
- Communications links between the sites
- BBUs at a TCR site
- Time code units at a TCR site
- Uplink hardware at a TCR site
- Downlink hardware at a TCR site
- Telemetry streams from the satellite

It is evident that there are many combinations of equipment units along the signal path from a ground system operator to the satellite and back. These combinations will be referred to as *data paths* in this paper.

THE PROBLEM

The system tester is faced with the challenge of showing that typical satellite operations can be performed with any of the data paths that an operator could choose. This has the added benefit of showing that single failures of equipment do not inhibit

satellite operations. However, demonstrating proper functionality is not the only reason for this testing. Exercising the different data paths also serves as an excellent checkout of the hardware and software during system integration and test. This provides a high degree of confidence in the system prior to formal testing.

Exhaustively testing all combinations and data paths is just not practical in today's fast-paced, hyper-competitive commercial marketplace. An efficient way of generating a minimal number of test cases that provide "good" coverage of the test domain is needed.

One solution is the combinatorial design method (CDM), as described in (Cohen et al. 1996).

COMBINATORIAL DESIGN METHOD

Simple Example. Consider the following example. You want to conduct a test of telemetry processing. You have several choices for the test configuration, depending on the TCR site, the BBU unit, and the satellite telemetry stream selected. Each of these parameters has two values. The TCR site can be Site 1 or Site 2. The BBU unit can be BBU 1 or BBU 2. The telemetry stream can be Stream 1 or Stream 2. An exhaustive test would consist of $2*2*2 = 8$ test cases.

Using the CDM can reduce the number of test cases by fifty percent (50%), as shown in Table 3. Only four test cases would be needed to show that any telemetry stream, any BBU, and any TCR site provide the capability to process telemetry data from the satellite.

Test Case	TCR Site	BBU	Stream
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

Table 3. Simple Test Case Example

In the CDM, the tester generates test cases that cover all pairwise, triple, or n -way combinations of test parameters (test factors). Covering all pairwise combinations means "that for any two parameters $P1$ and $P2$ and any valid values $V1$ for $P1$ and $V2$ for $P2$, there is a test in which $P1$ has the value $V1$ and $P2$ has the value $V2$ " (Cohen et al. 1996). The Telcordia Technologies (formerly Bell Communications Research) experience is that pairwise coverage is sufficient for good code coverage and checking the interactions of system functions (Cohen et al. 1996). Only pairwise coverage is considered in this paper.

This method also allows the tester to specify constraints on the test parameters. This prevents vacuous test scenarios or impossible relationships among the test parameters.

More Complex, Real World Example. The CDM can

be applied to complex systems, for which other methods are not practical (Huller 2000). Consider the following example from a recent project that the author worked on. There are four primary test parameters that represent redundant elements of the ground system. A fifth parameter, called FMRT, is included to test a special capability of the satellite. For all test cases, a common set of tests is run to demonstrate key functionality of the ground system. The test parameters and their values are shown in Table 4.

Test Parameters	Values
BBU Unit	Primary TCR Site - Primary BBU Primary TCR Site - Backup BBU Backup TCR Site - Primary BBU Backup TCR Site - Backup BBU
Real-time (RT) Server	SCC Primary SCC Backup SCC Remote
Ground Status and Control (GSC) Server	SCC Primary SCC Backup SCC Remote
Telemetry Stream	Stream 1 Stream 2
FMRT Testing	Yes No

Table 4. Test Elements for Ground System Example

Other considerations eliminated time code servers and RF equipment as test parameters for the redundancy testing. All test parameters are independent in this example.

Exhaustive testing would consist of 144 test cases. However, testing based on the CDM consists of only 12 test cases, or data paths, as shown in Table 5, Data Paths Configuration Table.

The table shows all pairwise interactions between the test parameters. Each value of a particular test parameter appears in a test case with every value of the other parameters. For example, consider the parameter value "Primary TCR Site - Primary BBU." It is tested with SCC Primary RT Server (test case #1), SCC Backup RT Server (test case #9), and SCC Remote RT Server (test case #10). It is also tested with SCC Primary GSC Server (test case #1), SCC Backup GSC Server (test case #9), and SCC Remote GSC Server (test case #10). Likewise it is tested with parameter value Telemetry Stream 1 (test case #1) and Telemetry Stream 2 (test case #9, or #10). And finally the parameter value is paired with (test case #1, or #10) and without (test case #9) FMRT Testing.

The table was generated by hand, based on the definition of pairwise coverage. Methods for

generating test matrices such as Table 5 are described later in the paper.

Data Path Number	BBU Unit	RT Server	GSC Server	Telemetry Stream	FMRT Testing
#1	Primary TCR Site - Primary BBU	SCC Primary	SCC Primary	Stream 1	Yes
#2	Backup TCR Site - Primary BBU	SCC Backup	SCC Backup	Stream 2	Yes
#3	Backup TCR Site - Backup BBU	SCC Backup	SCC Primary	Stream 2	No
#4	Backup TCR Site - Backup BBU	SCC Primary	SCC Backup	Stream 1	Yes
#5	Backup TCR Site - Primary BBU	SCC Primary	SCC Primary	Stream 1	No
#6	Backup TCR Site - Primary BBU	SCC Remote	SCC Remote	Stream 1	Yes
#7	Backup TCR Site - Backup BBU	SCC Remote	SCC Remote	Stream 2	Yes
#8	Primary TCR Site - Backup BBU	SCC Backup	SCC Backup	Stream 1	Yes
#9	Primary TCR Site - Primary BBU	SCC Backup	SCC Backup	Stream 2	No
#10	Primary TCR Site - Primary BBU	SCC Remote	SCC Remote	Stream 2	Yes
#11	Primary TCR Site - Backup BBU	SCC Remote	SCC Remote	Stream 1	No
#12	Primary TCR Site - Backup BBU	SCC Primary	SCC Primary	Stream 2	Yes

Note 1: The SCC remote server is at the backup SCC. The other servers are at the primary SCC.

Note 2: The data paths (matrix rows) have been reordered to facilitate testing of transitions between servers, and partly for convenience for test setup.

Table 5. Data Paths Configuration Table

TIME-TO-MARKET SAVINGS

The CDM is an innovative technique in hyper-competitive environments. It provides substantial savings in costs and schedule. This will be illustrated with the example shown in Tables 4 and 5. The CDM will be compared with a variation of exhaustive testing.

Background. All ground system projects conduct formal Factory Acceptance Testing (FAT) and Site Acceptance Test (SAT), which are witnessed by the Customer. The test group also performs dry runs of FAT and SAT. FAT is conducted in a controlled, laboratory environment at the factory; all equipment is in the same room. SAT is conducted at the Customer's facilities, which are dispersed among three sites as shown in Figure 1. Table 6 summarizes

time estimates for data paths testing, based on data collected for the ground system project.

Location	Time Estimates
Factory	First two data paths take 2.25 hours to test. Subsequent data paths take 0.50 hours, on average, to test.
Site	First two data paths take 2.50 hours to test. Subsequent data paths take 0.75 hours, on average, to test.

Table 6. Time Estimates for Data Paths Testing

The following assumption will be made: that a dry run takes as long as the corresponding formal test. In reality, the dry runs take 25%-40% longer,

based on the author's experiences, because of equipment configuration and test setup problems. However, this assumption simplifies the analysis below and makes the numerical comparisons below less artificial.

Table 7 identifies the personnel involved in either performing or monitoring the testing. Customer personnel are excluded from this analysis. For site testing, the primary test conductor and Lead systems engineer were at different SCC sites, to perform the testing. The support engineer and Lead systems engineer were needed to configure some of the equipment at the TCR sites.

<i>Test</i>	<i>Personnel</i>
FAT Dry Run	1 person: primary test conductor
FAT	3 persons: primary test conductor, Lead systems engineer, and Project Manager
SAT Dry Run	3 persons: primary test conductor, support engineer, and Lead systems engineer
SAT	4 persons: primary test conductor, support engineer, Lead systems engineer, and Project Manager

Table 7. Personnel Involved in Testing

The time needed to prepare the data-paths-specific test procedures for 12 data paths for FAT Dry Run/FAT was approximately 10 hours (plus an additional 6 hours for the procedures for the common tests that were run for each data path). The time needed to modify the data-paths-specific test procedures for SAT Dry Run/SAT was approximately 1 hour (plus an additional 2 hours for the procedures for the common tests). Only one person wrote and updated the test procedures. Management review time for the procedures is excluded from this analysis.

Quasi-Exhaustive Strategy. For the ground system example depicted by Tables 4 and 5, exhaustive testing would require 144 test cases, as noted earlier. It is very unlikely that project management would concur in running these 144 tests during FAT Dry Run, during FAT, during SAT Dry Run, *and* during SAT. A typical manager would point out that there are too many test cases, that there is not enough time in the schedule to run all the tests, and that the expenditure of labor does not seem cost-effective—especially when the ground system is not security- or safety-critical.

Instead, a lesser number of tests is needed. For this reason, a quasi-exhaustive strategy has been developed for the example—based on the author's engineering judgment. With this strategy, a tester would run the following:

- 100% of the tests (i.e., 144 test cases) for FAT Dry Run
- 10% of the tests, selected at random, for FAT
- 50% of the tests, selected at random, for SAT Dry Run
- 10% of the tests, selected at random, for SAT

More than a small subset of tests are selected for SAT Dry Run, for three reasons. First, a good checkout is needed to ensure that equipment is operational and was not damaged during disassembly at the factory (for packing and shipping), during shipping, or during reassembly at the sites. Second, an adequate checkout is needed to gain assurance that the ground system components will function properly in the Customer's environment. And third, a checkout must provide a level of comfort that the interfaces to Customer Furnished Equipment work satisfactorily.

Comparison. The schedule and labor effort required for the example in Tables 4 and 5 will now be contrasted for the CDM and the quasi-exhaustive strategy.

It is reasonable to assume that the time and personnel estimates described in Tables 6 and 7 for performing the data paths tests are the same for both strategies. It is further presumed that, after the test procedures for the initial 12 data paths for FAT Dry Run/FAT are developed, only 15 minutes, on average, are needed to tailor and generate the procedures for each additional data path test (by cutting and pasting parts of the initial test procedures). The time needed to modify the test procedures for the data paths for SAT Dry Run/SAT is assumed to be 1 hour for every 12 test cases, on average.

Time and labor costs for using the quasi-exhaustive strategy are provided in Table 8. Time and labor costs for using the CDM strategy are provided in Table 9. The labor costs reflect representative, fully burdened labor rates. The number of days for testing is based on 8 hours per day, for either executing test procedures or writing and modifying test procedures.

Table 10 compares the data from Tables 8 and 9, and highlights the significant benefits of the CDM strategy for the ground system example. Compared to a traditional company that would use the quasi-exhaustive strategy, an innovative company using the CDM strategy would reduce its system level test schedule by sixty-eight percent (68%) and save sixty-seven percent (67%) in labor costs associated with the testing. Being able to deliver a system faster and cheaper provides a significant market advantage to such an innovative company.

As noted in (Blackburn 1998), testing traditionally accounts for 40% to 75% of the lifetime

development and maintenance costs of a system. Reducing test costs will have a positive impact on a company's bottom line in a hyper-competitive business environment.

GENERATION OF TEST CASES

Manual Method. The CDM test matrix shown in Table 5 was generated manually. The following general approach was used. A partial example is provided to illustrate this approach.

Quasi-Exhaustive Strategy	FAT Dry Run	FAT	SAT Dry Run	SAT
Number of test cases	144	15	72	15
Time spent writing test procedures	49.00 hours	0.00 hours	8.00 hours	0.00 hours
Time spent performing data paths test procedures	73.25 hours	8.75 hours	55.00 hours	12.25 hours
Number of persons involved in testing	1	3	3	4
Total Labor Hours	370.5 hours (46.3 days)			
Total Cost of Labor	\$36,100 US			

Table 8. Resources for Testing with Quasi-Exhaustive Strategy

CDM Strategy	FAT Dry Run	FAT	SAT Dry Run	SAT
Number of test cases	12	12	12	12
Time spent writing test procedures	16.00 hours	0.00 hours	3.00 hours	0.00 hours
Time spent performing data paths test procedures	7.25 hours	7.25 hours	10.00 hours	10.00 hours
Number of persons involved in testing	1	3	3	4
Total Labor Hours	118.0 hours (14.8 days)			
Total Cost of Labor	\$11,900 US			

Table 9. Resources for Testing with CDM Strategy

CDM Strategy Compared to Quasi-Exhaustive Strategy	
Schedule Savings	68%
Cost Savings (Labor)	67%

Table 10. Time-to-Market Savings

1. Rank the test parameters by the number of possible test values, from highest to lowest. Ties can be decided arbitrarily.
2. Start with the test parameter with the largest number of values. (For the Table 4 and 5 example, this is the BBU Unit, with 4 values).
3. Then select a test parameter with the next largest number of values. (For the Table 4 and 5 example, this is either the RT Server or the GSC Server, with 3 values).
4. Then generate a matrix that has pairwise coverage for these parameters; this is equivalent to

- exhaustive testing for these two parameters. The number of resulting rows is a lower bound for the number of final test cases. (For the Table 4 and 5 example, this results in $4 \times 3 = 12$ test cases.)
5. Next consider a test parameter with the next largest number of values. Add a new column to the matrix—call it column K—corresponding to this parameter and its values. For column K, generate a set of values that provide pairwise coverage with the row values of the first column. If needed, add one or more new rows to the matrix. The author has found it helpful to vary the pattern in the column (e.g., consider using Y then X, instead of X then Y). It is OK at this step to leave some rows of the column blank; they will be filled in during the next two steps.
 6. Generate values to provide pairwise coverage between the new column (column K) and each of the preceding columns, starting with the second

column. That is, perform the following for each column J where J = 2, 3, ..., K-1:

Determine whether pairwise coverage exists between the row values of column K and column J. If not, try to use the blank (unspecified) rows in column K to generate the pairwise interactions. If this approach is not successful, it may be necessary to add one or more new rows to the matrix. Also consider modifying some of the values in column K to provide pairwise coverage between the row values in column K and column J. However, if any of these previously determined values are modified, it is necessary to recheck pairwise coverage with all the previous columns. The author's experience is that skill on this step is more a matter of art than science. Practice helps. Increment the value of J and repeat this step until pairwise coverage is achieved through column K-1.

7. If there are any blank values remaining in the rows of the column (column K), then arbitrarily fill them in. As a guideline, try to provide balanced coverage by having an equal number of occurrences of each value of the test parameter in the column. This may help with pairwise coverage if there are additional test parameters (columns) to be considered.
8. When done with column K, the values will have pairwise coverage with values in every other column.
9. Repeat steps 5 through 8 until all test parameters have been considered.

Partial Example of Manual Method. Consider the following example. There are three test parameters: P1, P2, and P3. P1 has four values A, B, C, and D. P2 has three values E, F, and G. P3 has two values H and I. Table 11 shows the partial test matrix after step 5 (K=3, J=1, using the terminology above). Note that in the last column, the pattern I-then-H is used sometimes, instead of H-then-I. This will help when doing step 6 (pairwise coverage of parameters P3 and P2). There are four blank, as yet unspecified, values in the last column of the matrix.

Test Case	P1	P2	P3
	A	E	H
	A	F	I
	A	G	
	B	E	I
	B	F	H
	B	G	
	C	E	H
	C	F	I
	C	G	
	D	E	I
	D	F	H

	D	G	
--	---	---	--

Table 11. Partial Test Matrix (After Step 5)

Automatic Method. The author is not aware of any free tools for automatically generating test cases with the CDM. However, Telcordia Technologies has developed a web-based application tool for the CDM, called AETGSM Web Service, where AETG stands for Automatic Efficient Test Generator. The tool produces test cases quickly and generates a near-minimum number of test cases (sometimes 2 to 3 more cases than the minimum number). Telcordia can be contacted to run efficiency algorithms that create a minimum set of tests. AETG Web Service allows constraints to be applied when test cases are generated. Particular combination of test parameters and their values that must be in test cases can also be specified. Telcordia is currently offering a free two-week trial of the AETG tool. Details and price information are available at the Uniform Resource Locator (URL) <http://aetgweb.argreenhouse.com>. The URL also contains hyperlinks to technical papers on CDM and experience with the AETG tool.

FURTHER INVESTIGATION

Additional research is needed to obtain a measure of goodness for the CDM strategy. The author poses the following questions. How much additional risk is incurred when exhaustive testing is not performed? What level of certainty does the CDM strategy provide for discovering latent bugs? How much of the test domain is covered by the generated test cases? Under what conditions is the CDM strategy not suitable? Can it be adapted for robust design of experiments (like Taguchi techniques)?

It would also be useful to develop case studies that compare the quantity and severity of Customer-reported problems (post-delivery) for systems using CDM with systems using exhaustive or quasi-exhaustive testing. The author has used CDM for the ground system described earlier, and has employed a CDM-like strategy (based on orthogonal arrays/Taguchi techniques) on two other ground systems. However, no valid, historical comparison data can be derived for these ground systems. A hyper-competitive business market will not easily allow such comparisons.

MANAGEMENT IMPLICATIONS

CDM offers management a way to reduce costs and schedule for the test program when there are multiple combinations of test parameters to be evaluated. Savings of approximately 70% for the ground system example have been shown. CDM can be applied whenever there are numerous data paths. It avoids the combinatorial explosion in the quantity of test

cases as the number of test parameters and values increases. It allows constraints or relationships among the test parameters to be accounted for.

During developmental testing, CDM can be used to generate a small subset of test cases. During operational testing, CDM can be applied to derive a small subset of operational scenarios or threads. Less testing is needed.

Being able to deliver a system faster and cheaper enables a company's management to beat its competitors to market.

CONCLUSION

In a hyper-competitive environment with constrained resources, the CDM provides an innovative approach for doing high-value testing with reduced costs and schedule. Based on the author's experience, CDM provides good coverage of the various data paths in a small, commercial satellite ground system, with a limited number of test cases. The method is flexible, and does not require training in statistics to use or understand. Managers and testers are urged to include CDM in their arsenal of systems engineering tools.

REFERENCES

- Aerospace Daily, June 21, 1999. This updates the article in Aerospace Daily, September 15, 1998. Aviation Week Group, McGraw-Hill Inc.
- Blackburn, M.R., "Using Models for Test Generation and Analysis." *Proceedings of 17th Digital Avionics Systems Conference (DASC)* (Bellevue, WA, October 31-November 7, 1998), IEEE, 1998, Vol. 1, pp C45/1-C45/8. Reprinted in *Software Tech News*, Vol. 3, No. 3, 1999, Department of Defense Data & Analysis Center for Software, Rome, NY, pp 11-16.
- Cohen, D.M., Dalal, S.R., Parelius, J., and Patton, G.C., "The Combinatorial Design Approach to Automatic Test Generation." *IEEE Software*, September 1996, pp 83-88.
- Huller, J, "Testing: Let Me Count the Ways." Unpublished, accepted for publication and presentation at the Tenth Annual International Symposium of the International Council on Systems Engineering (INCOSE), July 16-20, 2000, Minneapolis, MN.
- PRNewswire, C. E. Unterberg, Towbin, February 12, 1999.

BIOGRAPHY

Jerry Huller is currently a Senior Test Engineer at Raytheon Company in the Command, Control, Communication, and Information Systems business unit. He holds a B.A. in Mathematics from Davidson College and an M.A. in Applied Mathematics from the University of Maryland, College Park. Jerry has over

18 years of experience in defining system/software requirements, quality assurance testing, systems engineering, and customer technical support for military and commercial programs. For the past 4 years, he has served as Integration and Test Lead for several commercial satellite ground system projects. Jerry is a member of the INCOSE Verification and Validation Interest Group, and serves as Treasurer of the Colorado chapter of INCOSE.