**Introducing Combinatorial Testing to a Large System-Software Organization**

**Authors**: Jon Hagar, D. Richard Kuhn, Raghu N. Kacker

Affiliations: Lockheed Martin, Grand Software Testing, NIST Information Technology Lab

Publication: Submit to IEEE Computer

**Introduction**

This paper examines the introduction of combinatorial testing into a large aerospace corporation - Lockheed Martin (LM) [1]. In a market based economy, to stay competitive, organizations and the methods and tools they use must improve. When Lockheed Martin's interest in evaluating combinatorial testing (combinatorial test) started in 2009, combinatorial test was being used in pilot projects for US Air Force testing at the Eglin Air Force Base and was showing promise to reduce testing costs. Within the Department of Defense, the statistical Design of Experiments approach to testing was endorsed by the Office of Test and Evaluation in 2009 [2]. As discussed in more detail below, combinatorial test adapts DoE principles to software and aspects of system testing. A variety of commercial firms [3] had reported success with pairwise testing, which is a basic form of combinatorial testing that covers 2-way combinations of factors. Research by the National Institute of Standards and Technology (NIST) [4] had shown that 2-way testing can consistently detect 70% to more than 90% of faults, depending on the application. However, many DOD and Aerospace applications have much more stringent assurance requirements. For example, the FAA requires that life-critical systems have a failure probability of no more than $10^{-9}$. Testing and assurance costs for aerospace systems are very high, so approaches to reducing this cost while maintaining or improving quality are particularly valuable.

This paper looks at one improvement concept, the introduction of combinatorial test, which for the last few years has been under investigation at Lockheed Martin. The specific goal was to make available and start the use of combinatorial testing, without mandates from customers or management. To aid in these efforts Lockheed Martin developed a Co-operative Research and Development Agreement (CRADA) with NIST. As noted on the NIST web site [12], "CRADAs are partnering tools allowing federal laboratories to work with US industries, academia and other organizations on cooperative R&D projects. CRADAs provide flexibility in structuring project contributions, intellectual property rights, and in protecting proprietary information and CRADA research results." Specifically NIST provided tools, fixes to tool issues, training information, and reference material. This relationship continues in place today.

The driving factors to introduce combinatorial test included:

1) First to show the viability of the concept;
2) Test process improvement in a variety of domains: system, software, and hardware testing, which align with customer interest in improving testing ;
3) Make tests more effective in finding problems ; and
4) In the long run reduce the cost of testing or at least reduce the cost associated with the test lifecycle efforts by finding fewer errors late in development or worse, in the field.

Lockheed Martin entered into this investigation and improvement effort assuming it would take time, a series of efforts, and long term perseverance to introduce real change. This was done from an understanding of large organizations, what motivates them, and experience with process improvement efforts in the past. The effort was structured with an understanding that change takes time and evidence.

**Combinatorial Methods for Testing**

Among the test approaches that offer good fault detection with a small test set is a basic combinatorial approach called *pairwise testing* [3] in which all possible pairs of parameter values are covered by at least one test. Because system failures often result from the interaction of conditions that might be innocuous individually, this method can be effective for domains with many interacting parameters, such as most aerospace applications. Research by the NIST and others suggests that extended forms of combinatorial testing, covering combinations beyond simple pairwise, can be as effective as testing all possible combinations in some circumstances. Implementing this form of testing requires a *covering array*, which is a matrix that includes all *t*-way combinations of values for some specified interaction level *t*.

For example, suppose we want to test a 4-parameter function that inserts text into a document. Parameters and representative values for each are: location: {start, middle, end}, text_size: {small, medium, max}, document_size: {small, medium, max}, selection_method: {mouse, edit_menu, keyboard}, for a total of $3^4 = 81$ possible combinations to exhaustively test these values. Using the tests in Table 1, all 2-way interactions can be tested with only nine tests. That is, all possible pairs of values occur at least once among the tests.

| Test | location | text_size | doc_size | selection |
|---|---|---|---|---|
| 1 | start | small | small | mouse |
| 2 | start | medium | medium | edit_menu |
| 3 | start | max | max | keyboard |
| 4 | middle | small | max | edit_menu |
| 5 | middle | medium | small | keyboard |
| 6 | middle | max | medium | mouse |
| 7 | end | small | medium | keyboard |
| 8 | end | medium | max | mouse |
| 9 | end | max | small | edit_menu |

**Table 1. Pairwise test configurations**

A reduction in test set size from 81 to 9 is not that impressive, but consider a larger example: a manufacturing automation system that has 20 controls, each with 10 possible settings, a total of 10 x 10 x ....x 10 = $10^{20}$ (100 million trillion) combinations. Surprisingly, we can check all pairs of these values with only 162 tests [5], if the tests are carefully constructed.

**Empirical Basis of Combinatorial Testing**

What about the remaining faults? If 70% of failures involve just one or two parameters, how many failures will be triggered only by an unusual interaction of *more* than two parameters? This question has been investigated in a series of studies on software faults in a variety of domains. A study of failures in medical device software found that more than 95% of faults were triggered by either a single parameter or an interaction between two parameters [6]. Subsequent investigations found that many faults were caused by a single parameter value, a smaller proportion resulted from an interaction between two parameter values, and progressively fewer were triggered by 3, 4, 5, and 6-way interactions [7, 8, 4].

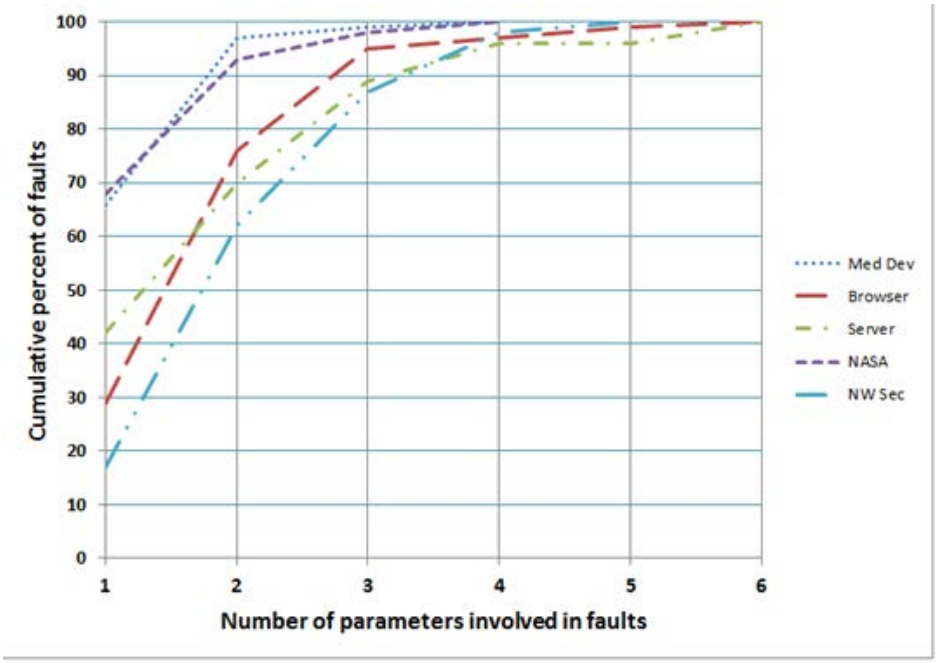


**Figure 1.** Percentage of errors triggered by *t*-way interactions

Figure 1 summarizes these results for the medical device study plus open source browser and web server applications, a NASA distributed database, and protocol vulnerabilities [4]. With the web server application, for example, roughly 40% of the failures were caused by a single value, such as a file name exceeding a certain length. Another 30% of the problems were triggered by the interaction of two parameters, and a cumulative total of almost 90% triggered by three or fewer parameters. Curves for the other applications have a similar shape, reaching 100% detection with 4 to 6-way interactions. While not conclusive, these results suggest that combinatorial testing which exercises high strength interaction combinations can be an effective approach to high-integrity software assurance. The key insight in combinatorial testing is this: *If we know from experience that t or fewer variables are involved in failures for a particular application type and we can test all t-way combinations of discrete variable values, then we can have reasonably high confidence in the application* [4].

**Combinatorial testing as adaptation of DoE methods** **[[sidebar]]**

Design of experiments (DoE), particularly in its variation known as "Taguchi methods", has been used for decades in industrial production with great success. But in its traditional form it was of limited utility for software testing. As originally used, DoE dealt with a handful of variables with a few possible settings each, but software testing may involve scores of variables, each with a huge range of possible values. DoE methods have been adapted for the special needs of software testing to the point where combinatorial testing is now practical.

DoE is a methodology for conducting controlled experiments in which a system is exercised for chosen test settings of various input variables (called factors). The corresponding values of one or more output variables (called responses) are measured to generate information for improving the performance of a class of similar systems. Conventional DoE methods were developed in the 1920s and 1930s by Sir Ronald Fisher and his contemporaries and their followers to improve agricultural production. Later DoE was adapted for experiments with animals, medical research, and then to improve manufacturing processes subject to uncontrolled variation. Frequently, the effects of many factors each having multiple test settings are investigated at the same time and the DoE plans satisfy relevant combinatorial properties [Box, Hunter, and Hunter (1978), Snedecor and Cochran (1991), and Montgomery (2004)]. The objective in conventional DoE is to improve the mean response over replications. Taguchi ( 1987) promulgated a variation of DoE methods for industrial experiments whose objective is to determine test settings at which the variation due to uncontrolled factors was least [Kacker (1985) and Phadke (1989)]. Taguchi promulgated use of orthogonal arrays (OAs), in which every factor-level combination occurs the same number of times.

With the advent of computers and telecommunication systems, independent verification and validation of software and hardware-software systems became important. Software engineers in various companies (especially Fujitsu in Japan and the descendent organizations of the AT&T Bell System in the US) started to investigate use of DoE methods for testing software and hardware-software systems. Use of OAs assured that all test settings for each test factor and all pairs of test settings for every pair of test factors were tested (executed). Thus began pairwise (2-way) testing, a type of dynamic testing in which the SUT is exercised for a test suite which satisfies the property that for every pair of test factors all possible pairs of test settings are tested.

Soon software testers realized the limitations of using OAs to construct test suits for software testing. Often, an OA matching the required combinatorial test structure does not exist. For example suppose four test factors have two test settings each and one has three test settings; thus the combinatorial test structure is $2^4 \times 3^1$. An OA of strength 2 matching the test structure $2^4 \times 3^1$ does not exist (it is mathematically impossible). In such cases a suitable OA can be modified to fit the need. Also, frequently, OA based test suites included invalid test cases (which cannot be executed). For example suppose in interoperability testing one factor is operating system (OS) with two test settings {XP, Linux} and another factor is browser with two test settings {Internet Explorer (IE), Firefox}. Since the browser Internet Explorer (IE) does not run on the operating system Linux, the pair {Linux, IE} is invalid. Since a test suite based on an OA includes all pairs of combinations the same number of times, this pair will be included in

some test cases and those test cases would be invalid. A consequence of omitting those test cases is that some valid pairs of test setting may not be tested. The limitations of the OAs led to the use of covering arrays (CAs) for generating test suites for software testing. It turns out that CAs have several advantages over OAs: (1) CAs can be constructed for any combinatorial test structure of unequal numbers of test settings. (2) If for a combinatorial test structure an OA exists (is mathematically possible) then a CA of the same or fewer test cases can be obtained. (3) CAs can be constructed for any required strength (*t*-way) testing, while OAs are generally limited to strength 2 and 3. (4) In generating test suites based on CAs invalid combinations can be deliberately excluded.

In combinatorial testing (unlike traditional DoE), the expected behavior of the system for each test case must be pre-determined. This requires additional information such as a mathematical model of the SUT or a benchmark implementation. In addition some tool is needed to check whether the actual behavior of the SUT matches the expected behavior and to make a verdict of passing or failing for each test case. Conventional design of experiments (unlike combinatorial test), are based on an assumed linear statistical model for the relationship between input factors and output response. In DoE, experimental error treated as random is an important contributor to the observed variation of response values. In combinatorial software testing, random error is absent, negligible, or can be avoided by appropriate modeling of the test factors.

**Combinatorial Testing Tools**

The primary tool used in Lockheed Martin's pilot studies is ACTS, a freely downloadable research tool [9] for generating combinatorial *t*-way test suites based on CAs with support of constraints (to exclude invalid combinations). Additional tools (described in a later section below) supplemented the capabilities of ACTS. Special features of the ACTS tool include the following:

(1) Excluding combinations of the test settings which are invalid according to user-specified constraints.

(2) Two test generation modes: scratch and extend. The former builds a test suite from the scratch, whereas the latter allows a test suite to be built by extending a previously constructed test suite which can save earlier effort in the testing process.

(3) Constructing variable-strength test suites. For example, of the 10 test factors all could be covered with strength 2 and a particular subset of 4 out of 10 factors (which are known to be inter-related) could be covered with higher strength 4.

(4) Verifying whether the test suite supplied by a user covers all *t*-way combinations.

(5) A Graphical User Interface (GUI), a Command Line Interface (CLI), and an Application Programming Interface (API). The GUI allows a user to perform most operations through menu selections and button clicks. The CLI can be more efficient when the user knows the exact

options that are needed for specific tasks, and is also very useful for scripting. The API interface facilitates integration of ACTS with other tools.

**Areas Considered for Combinatorial Test Application at Lockheed Martin**

Lockheed Martin development processes include integration and test activities as standard practices. Interest in combinatorial testing extends into systems, software, hardware, and materials testing. Specifically the following areas were of interest:

1. Application to the different configurations of systems, where a variety of user, hardware, electronic, and software options are often possible;
2. Use in software where there are ranges of parameters and variables which can interact to create errors;
3. Use in flight test scenario configurations where there can be many use conditions and hardware configurations, e.g. different targets, weapon options, altitudes, flight paths, or sensors for targeting;
4. Support for system-software User Interface testing where there can be many option combinations in menus, user displays, pages, and interactions; and
5. Application to hardware setting of complex systems, such as switch setting, sensor inputs, signals, and outputs, which must be tested.

The historic test processes focused on testing these areas, but often lacked systematic scientific approaches. Staff at Lockheed Martin quickly suggested others areas that might be considered. However to remain focused, the investigation concentrated on small initiatives designed to develop a basic understanding of combinatorial testing and associated implementation tools.

**Early LM Initiatives to Investigate Combinatorial Testing**

For some time the test technical staff at Lockheed Martin had known of the concepts of combinatorial test from various industrial sources. Early on in the combinatorial test efforts, LM contracted with some consultants to generate introductory training and informational materials, so that senior staff could get some exposure to the concepts. Combinatorial test was also included in an advanced class on software testing which was given to experienced testers. However, while testers "learned" about combinatorial testing, they only occasionally expressed interest in usage.

After four years, Lockheed Martin staff decided pilot efforts with funded commitment were needed. Lockheed Martin staff targeted one or two staff members to learn combinatorial test concepts and tools leading to R&D efforts. The R&D efforts were determined to include:

1) Conducting tool evaluation studies to determine which combinatorial test tools might fit Lockheed Martin's needs;
2) Some corporate funded case studies where aspects of ongoing projects were evaluated for the applicability of combinatorial testing;

3) Generation of an informational web site to provide knowledge and promote skills development; and
4) The creation of online set of web playable training materials with exercises so test staff can review and obtain the classes at any time.

Projects in large companies are often skeptical about any technology they are not currently using. This is because projects tend to be risk averse and new concepts such as tools that do not seem to have a sufficient track record may constitute a risk factor. To begin overcoming such risks, prototype evaluation studies can provide an avenue to assess a new concept while minimizing risks, particularly in this case since the funding of the study was done outside of project funding. Lockheed Martin conducted trade studies in the following areas:

1) Comparison to historic F-16 design and test problems [10]
2) Applicability to visual display testing
3) Use in Flight testing areas
4) Support of vehicle configuration (hardware and software) testing
5) Weapon configuration Testing
6) Test of a digital command system

For combinatorial testing to be truly effective, the use of software tools is recommended. As part of Lockheed Martin initial efforts, the company evaluated a series of tools. This was a decision analysis effort in which the tools were obtained, features compared, and tools applied to real problems. The tools evaluated included:

- Air Academy
    - Tools: SPC XL, DOE KISS, DOE PRO XL, DFSS MASTER
- Phadke & Associates
    - Tool: rdExpert
- Hexawise
    - Tool: Hexawise – web based software tool

*Note: Identification of certain commercial products in this article does not imply recommendation by NIST, nor does it imply that the products identified are necessarily the best available for the purpose.*

Each of these tools proved of value in the initial studies. Each tool was also found to have features of value depending on the context of the analysis (no one tool works perfectly on every problem). Finally, to provide a one stop place for combinatorial test information and tools, Lockheed Martin created and populated a web site.

**Results of combinatorial test efforts**

The focus of the study was to provide a technical expert to assess the method, learn/evaluate the tools, analyze data from possible pilot test plans/procedures, generate example cases, and encourage use. Some 15 areas were evaluated and eight pilots were executed in various business areas after corporate staff provided "educational" presentations. The pilot projects provide the following observations.

- Flight Vehicle Mission Effectiveness (ME) Test Area —
  - A series of 6 combinatorial test cases in different ME areas were created and worked with the existing test team, using historic ME plans.
  - The study compared combinatorial testing relative to test cases created from a statistical analysis tool. The results indicate that there is no major difference between the approaches, combinatorial testing vs. statistical, but the combinatorial tools were felt to be easier to use.
- Flight Vehicle engine failure modes test area
  - Failure combinations, accommodation, and fault tolerance combinations considered with constraints defined.
  - Historic plans analyzed and found to have "missed" cases with up to a 30% better coverage resulting from use of CT
- Flight Vehicle engine upgrade flight and lab testing
  - A total of 6 combinatorial test cases were generated covering the vehicle variations in various flight modes, and flight conditions
  - Generated cases compared favorably to existing team test plans
- F-16 Ventral Fin Redesign Flight Test Program [10]
  - The F-16 ventral fin redesign effort is a case study that demonstrates how combinatorial testing can be applied to design analysis problems, solutions, and evaluation
  - Generated a set of combinatorial test studies, data sets and reports which recreated the original analysis done, but using combinatorial testing instead of a highly experienced expert judgment (would have been a cost savings)
- Electronic Warfare (EW) system testing
  - 4 combinatorial test cases were generated, showing the validity of analyzing word based test plans for combinatorial test cases using a combinatorial testing tool
  - This effort showed the feasibility of one of the tool's features to "analyze" existing test plans

- Navigation Accuracy, EW performance, Sensor information, and Radar detection performance test points within system test flights
    - CT method and results showed the possibility of generating effective test cases in diverse subsystem interactions
- Electromagnetic Effects (EMI) Engineering —
    - An existing test plan was analyzed and new tests generated, but when compared to the existing plan, no improvements in combinatorial test coverage were noted, though it was felt the tool may have generated cases faster than original human analysis and judgment.
- Digital System Command testing
    - The system had sixteen commands that perform various file operations in three file systems. Most commands had more than one parameter, and some parameters accepted up to 3 values. Also, some parameters accepted a continuous range of integer values, so constraints were added to limit impossible operations
    - The effort produced 26 test cases spanning 1148 test combinations. The tests uncovered several major bugs that had gone undetected through unit testing

**Evaluation of Combinatorial Test Pilot Projects**

Beyond using only classic requirements-based testing, the combinatorial testing efforts demonstrated that the methodology had potential for effective and affordable design tests in software, hardware, system, and flight testing. Combinatorial testing proved applicable where multiple options/parameters are involved in the test structure.

CT does require some culture change for Systems Engineering (SE) and Test teams. The pilot effort showed that combinatorial testing must be used in upfront test planning, as well as having training and management support, to take full advantage of this approach across organizations. Our initial estimate is that this method supported by the technology can save up to 20% of test planning/design costs if done early on a program while increasing test coverage by 20% to 50%.

The tool investigation and pilot studies were generally positive with some future efforts identified. The combinatorial test tools that were investigated all worked, offered similar results, though each tool had features which might make it a viable selection for different criteria, e.g. open source tools were free if costs were an issue, some tools had better test plan analysis features, and yet others would work better where distributed teams were in use. Overall on the tool studies, Lockheed Martin found the combinatorial test tool environment maturing nicely.

Internal pilot efforts used a corporate staff resource who reviewed test problems on projects. Since the support was to be background, no mainline test efforts were impacted. General observations include:

- *Positive results*
    - Pilot study on F-16 showed how at cost SME effort could have been done with less costly staff
    - Several teams did pick up and use the tool with errors found.
    - Several of the investigated test problem spaces were shown to be viable.
    - These positive results start providing data points to engineers on the viability of the combinatorial test approach.

- *Mixed results*
    - Several areas where LM "analyzed" existing test plans, found weakness in test case coverage, but project did not like "adding" tests due to increased costs.
    - One project applied combinatorial testing, did testing and found at least one error, but felt the testing was not a vast improvement. However, they had no reference point for that view. Further investigation is underway.

- *Lesson learned*
    - Combinatorial testing was hard for teams to see where and how to apply to existing test efforts and practice. Many existing teams have "set" processes, and introducing new ideas can prove to be difficult unless a "champion" person takes on the task of promulgating combinatorial testing.
    - Some teams found the number of tests increased while they expected test savings, but this was because their tests were not providing good coverage in the first place. This was unexpected.
    - Teams who expressed interest failed to follow up due to constraints such as cost, skill, and/or time. In follow up efforts, the corporation has tried to supply training and tools to compensate for some of these areas.
    - Since many testers were not trained in techniques such as combinatorial testing, they see the tools as foreign to their experience and thus resist the concepts. Lockheed Martin has moved to provide online training and familiarization to start overcoming this resistance.

Web sites are the standard way many organizations preserve and present various forms of information. In the case of combinatorial testing, Lockheed Martin established a web site available to all employees with access to the intranet. The web site contains information such as: a basic introduction to combinatorial testing, information on related topics such as design of experiments, webinar materials, white papers, links to tools available within Lockheed Martin, external web sites, training, and presentation packages. The web site was promoted via internal company newsletters, presentations, and working groups. From its initial beginnings, the web site has continued to evolve and improve with continuing use today.

Finally, online training materials were created. These were based on the training packages and were enhanced with the knowledge received from the R&D efforts as well as groups like NIST. LM knew from several years of training in combinatorial testing and researching online classes that the material would

be best if presented in small modules with exercises and tool usage. These were constructed as prototype classes, made web accessible and loaded onto the combinatorial testing web site. The classes are in use today, though final more polished versions are planned using feedback from students.

**Future needs to support combinatorial test growth in industry**

The project results suggest that for combinatorial testing to gain more acceptance, the following effort should be considered. Approaches to integrating this technology into existing practice center on two themes: support materials including training packages and data on cost and effectiveness, and changes to not only tools but to the combinatorial approach itself.

***Support materials***

While supporting materials, tools, and information exists, there is room for improvement to speed the acceptance of combinatorial testing.

*Engineer educational guidance*

Guidance to help engineers learn and then determine when combinatorial testing may be useful in a testing problem is needed. This information would include:

- Text book and associated training classes at college and professional levels, particularly in online formats.
- Listing of examples in a "database" – Lockheed Martin has established within their combinatorial test informational web site a database where people can add their examples of where and how they have applied combinatorial testing. This can help others determine similar situations (learning by case examples). A future step might be for NIST or some industry group to establish a similar repository of applicable cases.
- Added industry web sites with more information, examples, presentations, white papers, and training materials.

*Customer and industry guidance*

Lockheed Martin functions in regulated and/or government environments. Standards could be updated to include:

- Industry standards – ISO 29119 software test standard defines and addresses combinatorial test.
- DOD – standards and/or policy guidance to use scientific test methods such as combinatorial test.
- Customer inclusion of scientific test techniques such as combinatorial testing in proposals, so all bidders play on a level playing field.

*Internal company guidance*

Internal Lockheed Martin efforts need to continue to promote the concepts and tools of combinatorial testing. Current plans include

- Basic training and a web forum now exist, but enhancements in the forum and more "hands on" training exercise can give more practical experience to the engineering staff.
- Lockheed Martin plans to establish a test design forum and community of practice where combinatorial testing and other approaches can be discussed and worked on. This can include the establishment of several subject matter experts and/or champions to help promote combinatorial testing on projects.
- Education of test management about the benefits of combinatorial testing by presentations within test leadership groups and at internal meetings.
- Support for project using combinatorial testing.

***Method and Tool Adaptations***

*Enhancing Tools*

Continuing advancement and enhancement of combinatorial test tools - The more robust and easy to use combinatorial test tools are, the more likely the approach is to be used. Also, the ability to integrate combinatorial test tool output with industry test tools/frameworks, e.g. an automatic import into Model based test tools, or incorporation with keyword based tools, will promote the use of combinatorial testing with these test tools which are gaining in popular use. Improvements should be applied to both commercial and open source tools.

*Adapting Methods*

It is not always practical to re-design an organization's testing procedures to use tests based on covering arrays. Testing procedures often develop over time, and employees have extensive experience with a particular approach. Units of the organization may be structured around established, documented test procedures. This is particularly true in organizations that must test according to contractual requirements, or relevant standards. And because much software assurance involves testing applications that have been modified to meet new specifications, an extensive library of legacy tests may exist. The organization can save time and money by re-using existing tests, which may not have been developed as factor covering arrays.

Short of creating new test suites from scratch, one approach to obtaining the advantages of combinatorial testing is to measure the combinatorial coverage of existing tests, then supplement as needed. Depending on the budget and criticality of the software, 2-way through 3-way or higher testing may be appropriate. Building covering arrays for some specified level of $t$ is one way to provide $t$-way coverage. However, many large test suites naturally cover a high percentage of $t$-way combinations. For example, it was found that tests developed for NASA spacecraft provided better than 90% pairwise coverage despite the fact that they were developed without the use of combinatorial testing tools [11]. If an existing test suite covers almost all $t$-way combinations, for an appropriate level of $t$, then it may be sufficient for the assurance that is required. Determining the level of input or configuration space coverage can also help in understanding the degree of risk that remains after testing. If 90% - 100% of the relevant state space has been covered, then risk is likely to be smaller than would remain after testing that covers a much smaller portion of the state space. These considerations led to developing an approach that involves measuring the combinatorial coverage of an existing test suite, then

automatically extending it to provide a desired level of coverage. A sophisticated tool was developed that measures coverage, while allowing for constraints (such as the typical example of excluding test configurations with the Internet Explorer browser on a Linux system).

**Summary**

The early efforts to include combinatorial test into the Lockheed Martin test culture have been positive albeit "slow" to be adapted. Areas and projects within the company have taken advantage of the tools, web site, and training. Individual engineers have been supportive and even enthusiastic. The traffic on the web site has grown, and interest in "improving" the prototype online training materials has continued.

The observation that existing projects with test plans in place have been slow to consider combinatorial test is likely an artifact of factors such as contracts and projects being "risk averse" to new ideas that were not contracted. We see new contracts and improvement effort going forward as being more likely to use combinatorial test. For this to happen, industry, company, and customer leaders must continue to advance concepts such as combinatorial testing.

Lockheed Martin is supportive of such improvement efforts and plans to continue the use of combinatorial testing, combinatorial test tools, our internal training and web sites, and upper management support. The company takes a long term view of such improvement efforts. The work reported here represents just the early steps in making combinatorial test a common practice in organizations such as Lockheed Martin. Lockheed Martin engineers look forward to working with industry, government, tool vendors, researchers, and others to promote combinatorial test.

**References**


1. Lockheed Martin Web Site http://www.lockheedmartin.com/us/who-we-are/organization.html
2. C. McQuery "Design of Experiments in Test and Evaluation". Office of Sec. of Defense, May 1, 2009.
3. J. Czerwonka, "Pairwise testing in real world: Practical extensions to test case generator", *Proceedings of 24th Pacific Northwest Software Quality Conference*, October 9–11, 2006, Portland, Oregon, USA, pp. 419–430, (2006).
4. D.R. Kuhn, D.R. Wallace, and A. Gallo, "Software Fault Interactions and Implications for Software Testing," *IEEE Transactions on Software Engineering*, 30(6): 418-421, 2004
5. C. Colbourn, http://www.public.asu.edu/~ccolbou/src/tabby/2-10-ca.html
6. D.R. Wallace, D.R. Kuhn, "Failure Modes in Medical Device Software: an Analysis of 15 Years of Recall Data", *International Journal of Reliability, Quality, and Safety Engineering*, Vol. 8, No. 4, 2001.
7. D.R. Kuhn and V. Okun, "Pseudo-exhaustive Testing for Software," *Proceedings of 30th NASA/IEEE Software Engineering Workshop*, pp. 153-158, 2006
8. D.R. Kuhn, M.J. Reilly, "An Investigation of the Applicability of Design of Experiments to Software Testing", *27th NASA/IEEE Software Engineering Workshop*, NASA Goddard Space Flight Center, 4-6 December, 2002 .
9. ACTS Combinatorial Testing web site. http://csrc.nist.gov/acts

10. J. Hagar, A. Cunningham, “A System Analysis Study Comparing Reverse Engineered Combinatorial Testing to Expert Judgment”, *IEEE Fifth International Conference on Software Testing, Verication and Validation* (ICST), 2012.
11. J.R. Maximoff, M.D. Trela, D.R. Kuhn, R. Kacker, “A Method for Analyzing System State-space Coverage within a t-Wise Testing Framework”, *IEEE International Systems Conference 2010*, Apr. 4-11, 2010, San Diego.
12. National Institute of Standards and Technology, Office of Technology Partnerships, Cooperative Research and Development Agreement, http://www.nist.gov/tpo/collaborations/crada.cfm. April 26, 2013.

References on DoE for sidebar

1. George E. P. Box, W. G. Hunter, and J. Stuart Hunter (1978) *Statistics for Experimenters*, New York: Wiley
2. George W. Snedecor, and W. G. Cochran (1991) *Statistical Methods*, Volume 276, New York: Wiley
3. Douglas C. Montgomery (2004) *Design and Analysis of Experiments*, 4th edition, New York: Wiley
4. Genichi Taguchi (1987) *System of Experimental Design*, Vol. 1 and Vol. 2, White Plains New York: UNIPUB, Kraus International (English translations of the 3-rd edition of Jikken Keikakuho (Japanese) published in 1977 and 1978 by Maruzen)
5. Raghu N. Kackar (1985) “Off-line quality control, parameter design and the Taguchi method,” *Journal of Quality Technology*, 17, pp 176-209
6. M. S. Phadke (1989) *Quality Engineering using Robust Design*, Englewood Cliffs New Jersey: Prentice Hall