# VERIFICATION "AUDITABILITY" CAN BE "FREE"

James H. Jones
Optants Documented Decision Support Company (ODDSCO)
297 Casitas Bulevar, Los Gatos, California 95032
www.optants.com consult@optants.com

**Abstract:** When simple audit support standards are applied to developing project requirements verification documentation in the current new product development environment, several benefits accrue. Data then made available from the verification documentation supports technical test reviews and an effective project status metric, as well as rigorously proving that verification is complete for specified requirements. Essential to verify requirements for systems affecting human safety, audit support standards enabled rigor could be useful during planning and testing to assure quality under ISO 9001 parameters claimed as met by many companies. When methods described in this paper virtually automate the resulting auditability, it essentially becomes "free".

## INTRODUCTION

### Definitions of Key Words:

***Auditability***. Key project documents are designed for either providing or contributing to provision of an audit trail for accomplishment. The traditional trace matrix reveals allocations from system requirements to their subsidiary specification requirements. Auditability for system engineering tasks requires traceability, at least beyond that early requirements management task of allocation, to the project back end for verification and validation (V&V). Auditability improves further if the V&V tests identify requirements addressed therein and has most rigor when annotated at individual steps.

***Permutation*** (with reference to a requirement). Among all possible occurrences of a function that is specified by a requirement statement for the system at hand, one of the distinct and meaningfully separate instances that must be part of any comprehensive verification. Some requirements are verifiable with a single measurement, but developing a provably complete system verification means identifying all requirements permutations.

***Requirement.*** A natural language statement to specify a mandatory characteristic or some functional behavior to be provided by a product or process. System level requirements are the initial translation of product functions into engineering terms and should not define implementation for other than required interfaces to other systems. Subsidiary level requirements result from allocating appropriate portions of each system level requirement to the electronic, mechanical, and software subsystems comprising the architecture. Requirements can describe a proposed product such that any competent entity can be contracted to design and build it. As implemented by the design engineers, the ultimate incarnation of the product must fulfill the allocated requirements.

***Validation***. Determining "the right system was built" for the defined customer(s). The market is the ultimate validation for any system. System developers attempt to approximate the customer environment and define qualification requirements for testing to which systems are subjected before delivery readiness may be declared. Verification is a subset of the overall system validation.

***Verification***. Determining "the system was built right" or in conformance to its specifications as allocated from the system requirements to mechanical, electrical, and software requirements. Correct fulfillment is verified by instrumented test, demonstration, inspection, analysis, or a combination therof (when some permutations defy using the same method).

***Verification Audit Support Standards***. Requirements Management methods (classical Systems Engineering) provide standards that automate the auditability of requirements verification. Document templates (format rules and content descriptions) for requirements specifications are the foundation for all subsequent auditability. Similarly, a document template for the trace matrix provides specific audit support to the requirements verification. Finally, adhering to special standards during development of test documentation provides comprehensive rigor to overall traceability for the desired ultimate auditability.

**Application.** Simply put, the audit support standards are analogous to the software source code comment standards that require stating both what and why for each segment developed. Such extra thought and work enables quicker learning by persons new to complex systems as well as rapid reengineering for maintenance of legacy systems. Spending such thought and work on each requirement, to amplify traceability with the audit support standard notation shown in this paper, will provide the following downstream benefits:

1. Assistance to confirming propriety of requirements derivation from system level/product requirements.

2. Support for a project metric of Percent Complete per requirement and higher level of SRS hierarchy; easily updated as test development proceeds and when any requirements change.

3. Assistance to detailed technical review of planning and procedural implementation of verification for each requirement.

4. Support for verification development audit at any time during the task, fulfilling the ultimate goal.

**System Engineering Process Fundamentals.** System Engineering 101 uses FRAT as the acronym for phases of New Product Development. In brief:

***Functions*** are set forth for the proposed new product by Marketing or some other surrogate for the customer. No matter how well defined, functions are nevertheless not adequate to actually design and build a product. So, in a typical Concept Design stage:

***Requirements***, developed via the System Engineering process of analyzing and translating the proposed set of functions, form the engineering specification for design criteria and product performance.

***Architecture***, illustrated by a functional block diagram plus software diagrams for embedded processor(s), is the general concept for a product incarntation intended to fulfill the specified requirements.

***Testing*** of prototypes and final product configurations, often known as Verification and Validation (V&V), is necessary to confirm or deny requirements fulfillment.

In other words, once specified, "Requirements Rule!"

Accordingly, a typical development process includes a Requirements Traceability Matrix (RTM) that shows assigned allocations from the system level specification requirements to subsidiary specifications requirements.

It is a top-down listing, with the tabulation ordered in accordance with the system level specification table of contents. The columns are labeled for: ID (paragraph number or other requirement identifier), System Level Requirement Text, ID, Subsidiary Level Requirement Text, and Comments. Each row contains a separate allocation, as the one-to-one (or part of a one-to-many) derivation from the system level requirement.

The system (product) specification level is where the impact of insufficient rigor is greatest upon the entire project life cycle. Audit support standards apply to the design qualification requirements therein as helpful to the Validation processes. Nevertheless, audit support is most useful to Verification of functional performance requirements allocated to the subsidiary level (such as the software/hardware requirements specifications).

**The Normal Process.** Despite frequent lack of explicit recognition or any comprehension of its value in many commercial entities, some of the System Engineering processes will occur by default. In any case, one of the major processes, Requirements Management, should be applied deliberately and early in a development project because well written requirements are seldom the norm. (That assertion soon will be addressed.)

The simple RTM can be made into a management tool, but it requires substantial expansion to add that value. Requirement testers expand RTMs to include a column just prior to Comments for identifying V&V test(s). It lists procedures that, when all have been executed, fully verify each subsidiary requirement. Figure 1 shows the deliverable table format (usually landscape).

**Figure 1**

| [PRODUCT] REQUIREMENTS VERIFICATION TRACE MATRIX (Format) | | | | | |
|---|---|---|---|---|---|
| **PS ID** | **System Requirement** | **SS ID#** | **Allocated Requirement** | **VP ID#** | **Comments** |
| 3 | Functional Requirements | S_3 | Functional Requirements | N/A | Header Label |
| 3.1 | Subsection 1 Title | N/A | N/A | N/A | Header Label |
| 3.1.1 | Category 1 Title | N/A | N/A | N/A | Header Label |
| 3.1.1.1 | System shall (requirement 1) | E_3.4.7.3 | CPU shall (provide function) | E_3.1.2 | |
| 3.1.1.1 | | S_3.3.2.1 | SW shall (provide function) | S_3.3.14 | |
| 3.1.1.2 | System shall (requirement 2) | M_3.3.2.1 | SW shall (provide function) | S_3.2.1 | |
| ... | ... | ... | ... | ... | ... |

It is most useful, during testing development, to sort the RVTM in order of the requirements in subsidiary specifications. In addition to a more "natural" grouping of requirements subject to verification test, adequacy of system level requirements or necessity for their revision (as drivers for the subsidiary requirements) becomes apparent. A subset of the reordered RVTMs is in the domain of requirement verifiers. It is like the four rightmost columns of Figure 1, but lists the subsidiary specification hierarchy titles as well as all the requirements specified therein. See Figure 2 for the Subsidiary Requirements Specifications (Electrical, Mechanical, and Software) RVTM format.

To complete the description of a normal Requirements Management fulfilling approach, every Test Procedure must list the ID of every requirement (and desirement) tested therein. Often, this is the extent of the tracability provided, but is it enough? Perhaps.

If requirement testers are diligent and list requirement IDs in test procedures only after all planned verification is implemented therein, audits and testing adequacy reviews can be accomplished. Still, difficulties abound. Until shown exactly where a selected requirement is being tested and to what extent, the suspicion justifiably exists that its listing is merely an assertion. Procedure authors may not be available so some extensive reverse engineering by an "appointed" (lucky?) engineer then is necessary to support the audit. Because that really is not an effective use of their time, explanation of a much better method begins here.

**Figure 2**

| SUBSIDIARY (ERS/MRS/SRS) REQUIREMENTS VERIFICATION TRACE MATRIX (Format) | | | |
|---|---|---|---|
| **ERS ID** | **ERS Requirements** | **VP ID#** | **Comments** |
| 3 | Functional Requirements | N/A | Header Label |
| 3.1 | Subsection 1 Title | N/A | Header Label |
| 3.1.1 | Category 1 Title | N/A | Header Label |
| 3.1.1.1 | The [Electrical item] shall (requirement 1) | E_1.3.3 | |
| 3.1.1.2 | The [Electrical item] shall (requirement 2) | E_3.2.4, E_4.7.12 | (2 test procedures are used for coverage) |
| ... | ... | ... | ... |
| | | | |
| **MRS ID** | **MRS Requirements** | **VP ID#** | **Comments** |
| 3 | Functional Requirements | N/A | Header Label |
| 3.1 | Subsection 1 Title | N/A | Header Label |
| 3.1.1 | Category 1 Title | N/A | Header Label |
| 3.1.1.1 | The [Mechanical item] shall (requirement 1) | M_3.3.2 | |
| 3.1.1.2 | The [Mechanical item] shall (requirement 2) | M_5.3.1 | |
| ... | ... | ... | ... |
| | | | |
| **SRS ID** | **SRS Requirements** | **VP ID#** | **Comments** |
| 3 | Functional Requirements | N/A | Header Label |
| 3.1 | Subsection 1 Title | N/A | Header Label |
| 3.1.1 | Category 1 Title | N/A | Header Label |
| 3.1.1.1 | The [Software item] shall (requirement 1) | S_3.3.2 | |
| 3.1.1.2 | The [Software item] shall (requirement 2) | S_3.3.1/4/7 /14 | (4 test procedures are necessary for coverage) |
| ... | ... | ... | ... |

## VERIFICATION AUDIT SUPPORT STANDARDS

Providing adequate audit and detailed review support requires making some integrated changes to individual requirements, to Test Procedures, and to the Subsidiary Requirements Verification Trace Matrix (SRVTM) just mentioned.

**Requirements Specification.** Fundamental to concepts of Requirements Management is Formal Specification Language for individual requirements statements text. That is, when simple stylistic rules are used, several benefits follow. The important three among them are that ambiguity is reduced, categorization is easier, and verifiability is improved (which reviewers and auditors also appreciate).

What are the simple, stylistic rules for requirements?

First, although they should be be written in the same style because they swap places with priority changes, differentiate the requirements from "desirements." (A popular convention is to require the word "shall" in all requirements and the rule is to develop only that which is required because the still unsurpassed definition of quality is: "conformance to requirements" [Crosby].)

Second, limit requirements to one function each, unless the list is of closely affiliated items, such as named variations of essentially the same general function.

Third, avoid implicit requirements by always naming the subject of the specification before "shall."

Fourth, avoid "shall not" wording that requires testing the universe to prove a negative. That is, to confirm the nonexistence of a specified thing. (Use "shall inhibit" or similar words that can lead to some positively verifiable implementation.)

Fifth, avoid the redundancies of "be capable of" or "be able to" by using a direct action verb statement.

Finally, be cautious with adverbs (verb modifiers) or adverbial expressions and, whenever possible, replace all adjectives (noun qualifiers) with measurable values, ranges, or limits.

From a System Engineering perspective of auditability, the reason should be obvious:

***Until a specification statement can be fully verified by any of instrumented test, demonstration, inspection, or analysis (or a combination thereof), it is not a valid requirement.***

That assertion implies that requirement testers should have veto power over specifications content (especially when they cannot be involved early in the System Engineering process and are limited to reviews.)

**A Requirements Statement Audit Standard.** When the entire Integrated Product Team is familiar with use of Formal Specification Language, requirements tend to be verifiable as stated. Generalized, the standard form for a requirement statement is:

*[Optional qualifier,] The (system/subsystem/process name) shall ["should" or "will" if a desirement] (direct action verb statement of function) [, optional qualifier]*.

When every requirement follows the generalized form and rules just set forth, Requirements Management can extend auditability by clearly separating functional and design qualification requirements into differing major document sections. (The Data Item Descriptions that are followed to fulfill Department of Defense contracts require this.) Verification and Validation are thereby differentiable by major sections in the specifications. Thus, when different organizations are tasked to do each "V" type, requirements fulfillment confirmation "ownership" is known.)

To further apply Requirements Management, develop and apply Document Templates that formalize the rules set forth for all project documents, such that each will become part of a tightly integrated system. Design organizational format and content of each to fulfill the worst case requirements of all internal and external customers for specific sections and subsections of plans, requirements and specifications. Then, templates help provide extensive support for any Product Assurance reviews or regulatory agency audits.

The templates thereby become very useful for future new product development projects because the detailed content descriptions clearly support estimation of the resource needs for documentation development tasks.

When a project is less than worst case, the total effort should be reduced. Nevertheless, to fulfill the general standards of ISO 9001 and regulatory oversight audits, we must retain ability to show that project work was planned and remains in accordance with that plan.

Accordingly, develop the Project Plan (or have it cite a separate tailoring plan) to reduce the template planned content of each document appropriately to match the new project's reduced scope and scale.

For processes projects, only the system specification is generated. The resulting sequence may undergo some formal validation process, if deemed necessary for its application. Documentation of a development project for an intangible process also can become the subject of regulatory audits.

**Test Procedures.** Assumption is that the requirement testing is divided into categorical groupings of test cases and subsidiary test procedures. Subdivision of the suite of formal tests is common, to enable simultaneous testing with multiple testers and qualification systems. Even when the equipment is expensive, very large, or otherwise unsuitable for applying simultaneous testers, subdivision of the testing procedures enables regression testing (using a subset of the previously defined testing) following inevitable minor changes.

One recommended approach to test procedures format can be helpful to both auditors and reviewers even if the integrated SVRTM change is not implemented. For manually performed testing, it assumes a tabular format in the body of the document with a column for test number, a column for setup process, a column for expected result, and a column for pass/fail registry as in Figure 3. Within each step assigned to test one or more of each requirement's permutations, simply annotate each expected result item with both the requirement ID and extent of the testing to be accomplished.

For reviewers and auditors information, whenever other permutations of the same requirement are tested in the same procedure, place the total number of such tests in parentheses after its ID in the header listing.

Similarly, the necessary annotation can be placed in automated test scripts, test program source code with comment statements, or in the test results.

Now, after a quick explanation and example search by test procedure author or other individual familiar with audit support annotation, a reviewer or auditor should be able to find all the claimed testing without additional help. Determining technical adequacy of the testing for each permutation may require relevant knowledge, but the facilitated entry level of provided audit or review support has become determination of claimed testing plausibility by non-technical persons.

This means that an inspection by quality assurance can be for much more than apparent logical flow, spelling, and correct numeric sequence.

**Figure 3**

| **REQUIREMENTS VERIFICATION TEST PROCEDURE (Format)** | | | |
|---|---|---|---|
| **Test** | **Test Setup and Measurement(s) Process** | **Expected Results Descriptions** | **Actual Results** |
| **1** | **Initialization** | **N/A** | **N/A** |
| 1.1 | Setup to measurement | Expected result for measurement 1.1<br>(Part of S_3.1.1.2 Parameter 2) | **P F** |
| **2** | **Test Category Title** | **N/A** | **N/A** |
| **2.1** | **Subcategory Title** | **N/A** | **N/A** |
| 2.1.1 | Setup to measurement, requiring recording a value and comparison | Value _______________<br>Expected result for measurement 2.1.1<br>(Part of S_3.2.1.3 Parameter 1) | **P F** |
| 2.1.2 | Setup to measurement, requiring recording a value and comparison | Value _______________<br>Expected result for measurement 2.1.2<br>(Last of S_3.2.1.3 Parameter 4) | **P F** |
| **2.2** | **Subcategory Title** | **N/A** | **N/A** |
| ... | ... | ... | ... |

**SRVTM Upgrade.** One part of the recommended audit support upgrades is most helpful to requirements testers (the test procedure authors) but assists the examinations by any auditors and reviewers. Insert a Permutations column in the SRVTM just after the VP ID# column, to hold a list of the permutations identified for each requirement. Divide the list into permutations with testing defined (as listed in the adjacent column) and those for which test development remains incomplete. Integration of this update requires that the described Test Procedure annotations for permutations addressed be clearly linked with the SRVTM permutations list.

The permutations listings assist technical review of the test planning and implementation for each requirement. As stated in the Introduction Benefits list, the review or audit can occur at any stage in the testing development and still be useful because it helps confirm for ISO 9001 or regulatory auditors that all project work has been performed in accordance with planning.

Testing development which remains incomplete is easy to identify by scanning the Permutations column of the SRVTM. (If this trace matrix is a document, search for the next "(To do)" would find them quickly.)

**Figure 4**

| SUBSIDIARY SPECIFICATION REQUIREMENTS VERIFICATION TRACE MATRIX (Format) | | | | |
|---|---|---|---|---|
| **SS ID** | **SS Requirement** | **VP ID#** | **Permutations List** | **Comment** |
| 3 | Functional Requirements | N/A | | Header Label |
| 3.1 | Subsection 1 Title | N/A | | Header Label |
| 3.1.1 | Category 1 Title | N/A | | Header Label |
| 3.1.1.1 | The [subsystem] shall (requirement 1) | | Permutation 1, Permutation 2, Permutation 3 (To do) | (Probably all in ?_3.3.2) |
| 3.1.1.2 | The [subsystem] shall (requirement 2) | ?_3.3.1.1/4 /7/14 | Permutation 1, Permutation 2, Permutation 3 (Done)<br>Permutation 4, Permutation 5, Permutation 6, Permutation 7 (To do) | (4 VPs necessary for coverage) |
| ... | ... | ... | ... | ... |

**Percent Complete Metrics Support.** This last part of the recommended upgrade responds to management's modern mantra that metrics are mandatory. Alliteration aside, that means you must come up with a metric that reports something of use to you or your management is likely to invent a metric that doesn't. Worse, assuming unintended side effects are only neutral or benevolent rather than actively counterproductive, developing data to support reporting it could require extensive effort.

Therefore, whenever you must expend effort to develop and process data to report an assigned metric, anyway, a more useful or cheaper metric can be thought of as essentially free.

One established principle is that the "best" metric is a byproduct of what you already should be doing to help yourself or to fulfill a work requirement. Accordingly, the Percent Complete metric is ideal for reporting status of the V&V development effort.

However, implementing this metric requires yet another substantial change to the defined SRVTM:

Insert two new narrow columns before the Permutations column, as shown in Figure 5. Label the first as TP, for total permutations, and the second as TD, for tests done. The TP column is filled with its number during verification planning for each subsidiary level (SRS) requirement or desirement. The TD column is updated upon completion of the individual permutation testing development within the planned Test Procedure(s).

Another time for update(s) is when requirement change adjusts the parameters count and associated testing.

The Percent Complete for each individual requirement is computed from the TP and TD entries as follows:

$$\text{Percent Complete} = TD/TP * 100$$

Those who have been paying attention, thus far, know that the numbers in TP and TD are redundant to simply counting and calculating from the previously listed groups in Figure 4 Permutations column (to maintain testing development status).

**Figure 5**

| SUBSIDIARY REQUIREMENTS VERIFICATION TRACE MATRIX (Format) | | | | | | |
|---|---|---|---|---|---|---|
| **SS ID** | **SS Requirement** | **VP ID#** | **TP** | **PD** | **Permutations List** | **% Complete** |
| 3 | Functional Requirements | N/A | 7362 | 1106 | | 15 % |
| 3.1 | Subsection 1 Title | N/A | 184 | 157 | | 85 % |
| 3.1.1 | Category 1 Title | N/A | 33 | 12 | | 36 % |
| 3.1.1.1 | The [subsystem] shall (requirement 1) | | 3 | 0 | Permutation 1, Permutation 2, Permutation 3 (To do) | (Probably all in ?_3.3.2) |
| 3.1.1.2 | The [subsystem] shall (requirement 2) | ?_3.3.1.1 /4/7/14 | 7 | 3 | Permutation 1, Permutation 2, Permutation 3 (Done)<br>Permutation 4, Permutation 5, Permutation 6, Permutation 7 (To do) | (4 VPs necessary for coverage) |
| ... | ... | ... | | | ... | ... |

While only you may be interested in tracking the status of V&V development for each individual requirement, the two new columns are invaluable for collecting and reporting the Percent Complete metric at higher levels. As shown in Figure 5, the newly expanded SRVTM format assists maintaining the data for that metric.

Further, although it still contains notes, the rightmost column has been renamed to “% Complete” to better illustrate the upcoming explanations.

Because each SRVTM lists only the requirements for an individual specification, it should include the entire hierarchy of the document in that ordering. The TP and TD cells in the header label rows for each group of requirements contain the totals of permutations for each group. The Percent Complete values at this level are likely to align with entries in the detailed project schedule and, unless micromanagement is rampant, be the lowest level at which that metric is computed and reported. (Reporting at an individual requirement level is likely to occur only once or twice before higher level reported values are trusted, but insert a new column for the metric when micromanagement is anticipated.)

Similarly, the TP and TD cells in the header label row designating the parent of each category of requirements (major subsection header row) will contain the sum of all that subsection’s category label rows totals. Percent Complete metric values represent a major verification testing task status, appropriate for reporting at program management levels.

Finally, the TP and TD cells in the header label row designating the parent of all requirements (Functional Requirements header row) will contain the sum of all subsection’s label row totals. This Percent Complete metric value represents total requirements verification or validation testing for an entire specification, so is useful for summary reporting of status at the senior management level.

If hardware development teams are similarly required to report the status of their formal testing development, the value of tests required and tests developed columns in the electronic and mechanical design versions of the SRVTM is obvious.

For automated testing, the appropriate metric elements can be computed during execution of the test program. Inclusion in test results files is appropriate for reporting during program development (but commented out, until needed following a major requirements change).

## SRVTM WITH A SPREADSHEET

While the project RVVTM should be produced with word processing, because it contains a preponderance of text as part of audit support, customers for various SRVTMs are more specialized toward verification and validation. The tabular text can be minimal. So, of the columns of Figure 5, only the first and last four need be used (plus any metric column). Then, a cheapseats but highly effective approach is a spreadsheet SRVTM.

The requirements level rows still need manual input of permutation total and done counts in TP and TD cells. The aligned metric cell contains the Percent Complete formula for that row

The third level ID TP and TD cells sum the range of their subordinate requirement level cells and their metric cell for each row contains the Percent Complete formula.

Similarly, the second level ID TP and TD cells sum their subordinate group third level cells and their metric cell contains the Percent Complete formula.

The first level ID TP and TD cell sums its subordinate group second level cells and its metric cell contains the Percent Complete formula.

## RELATED METRICS

The SRVTM word processing table or spreadsheet can be expanded by new columns for count of permutations done and the associated Percent Complete metric for each milestone used in their V&V development. Two candidates for addition are Design Done and Dry Run.

## OTHER SUPPORTING TOOLS

Another approach is to use a specialized database tool for the Requirements Management system to provide custom reports for the RVVTM and SRVTMs. Today, a selection of such tools exist, but with widely differing capability and prices per seat. RequisitePro® is at the moderate end, while Doors® and Slate® are higher end. A Web search for "Requirements Management" will result in hits for many such related tools.

## AUDITABILITY INSERTION

When being in crisis recovery mode prevents doing the entire job, determine process improvements that could be implemented within the time constraints. Updating specifications format probably should await the start of a major project segment. Retaining existing identifiers doesn't preclude rewriting requirements into formal specification language. Alone, that improves clarity of RVVTM traceability, identification of the requirements permutations, and simplifies the eventual specification reformatting. Retrofit of test procedures to indicate where each permutation is verified/validated may take longer than the other tasks, because of necessity for repetitive scanning through many steps. Nevertheless, insertion of audit support standards does establish proof of comprehensive rigor to the extent it is completed.

## CONCLUSIONS

Application of verification "auditability" is shown to be part of a systematic Requirements Management process that is likely to reduce time and effort (and, thereby, cost) over the product development cycle. When the methods application to your requirements verification and validation will provide comprehensive rigor at less cost than the previous practices would have, claiming that auditability is "free" is not exaggeration. [Crosby] Even when the product does not involve human safety, professional pride is supported by the increased rigor. (Asserting how well your requirements V&V is done is not boastful anytime, if you easily can prove it.)

## REFERENCES

Crosby, Philip B., *Quality is Free*, McGraw Hill, New York, 1979

## BIOGRAPHY

**James H. Jones** is Owner/Principal of ODDSCO (Optants Documented Decision Support Company), a consulting business he founded to provide practitioner level seminars and educational material on Project and System Engineering processes adapted from a Mil-Spec approach to suit small businesses and commercial new product developers where inexpensive, effective tools are essential for competitiveness (or sheer survival).

He received a B.S. in Business Administration from College of Notre Dame at Belmont, CA, and his M.S. in Industrial and Systems Engineering from San Jose State University, CA.

He gained skills in comprehensive, rigorous software requirements verification of embedded systems during work for defense corporations (Trident I submarine-launched missile Flight Control Computer, targeting system for the Tomahawk antiship cruise missile, and several combat aircraft tactical display processors). His most recent embedded systems requirements testing experience is with medical device (IV infusion pumps) software involving human safety and compliance with FDA regulations.