

An Interrelated Approach to Requirements Management

David M. Eiband

Many of us have imagined this one: the Service needs a new system to fill an operational need; you meet with the users and gather data, put together your team, produce the required documents, satisfy the required reviews—and, voilà, you've developed, tested, and fielded the essential system. "How tough can this be?" you ask.

The real world, unfortunately, is not so straightforward and not nearly so forgiving, and many of the problems that we face in acquisition find their genesis in the very first actions on a program, especially when dealing with requirements. This article will discuss some potential approaches to ease treatment of requirements across a comprehensive, full-up program.

Setting the Stage

Before jumping into solutions, some examination of "requirements" is warranted. We all know that the JCIDS—Joint Capabilities Integration and Development System—establishes a process to identify and validate solutions to capability gaps. The products of that process germane to this discussion are the Initial Capabilities Document (ICD) and the Capability Development Document (CDD), but later-changing requirements would also be included in the Capability Production Document (CPD). If the ICD and CDD contain the capability requirements, are there any other requirements necessary to efficiently conduct a program?

The answer to that question is unequivocally "Yes." As the systems engineering process develops, a design so-

lution will lead to many more technical requirements in addition to the capability requirements noted in the ICD and CDD. And in addition to purely technical capability requirements, other requirements will develop in such areas as operational site construction; industrialization; construction, conversion, or expansion; or equipment modernization. Most programs will also generate requirements for quality assurance, first article testing, or lot acceptance. Nontechnical requirements will also arise. A properly structured program will have requirements for program, systems engineering, and risk management programs. The program office will also establish requirements for configuration, data, and interface management programs. Finally, most programs would identify requirements for cost control systems.



Many of the problems that we face in acquisition find their genesis in the very first actions on a program, especially when dealing with requirements.

Eiband is professor of systems engineering with DAU. He earned his bachelor's degree at the U.S. Air Force Academy and his master's degree at the University of Texas at Austin.

One can only conclude that there are a lot of requirements floating around any program. This indicates that we must do several things: first identify all appropriate requirements; second, maintain those requirements; and third, control any changes to the requirements set. Fortunately, there are several approaches and tools available to assist us in this effort.

The Big Picture

While often the butt of jokes, the Big Picture is a usually a nice place to start on most projects. In our case in acquisition, our end product will never be delivered for use until we accomplish certain activities. First, we should almost always presume that some significant portion of the work will be done on contract; and to accomplish a contract, we will always need a Statement of Work (SOW), we will more than likely need a specification, and we will also need a Work Breakdown Structure (WBS).

Luckily for us, a lot of smart people over many years have developed procedures and tools that will greatly improve our chances for success. MIL-HDBK-245D, *Handbook for Preparation of Statement of Work*, provides clear instructions for writing a SOW or a Statement of Objectives if that approach meets your acquisition's needs. The *Handbook* clearly and succinctly defines the relationship between the performance requirements properly located in a specification, the non-specification work performance requirements located in the SOW, and the proper method for the order and delivery data. Given that there are only three deliverable products from any government contract—a technical product, non-technical products/services, and data—the *Handbook* is a very useful tool. In addition, the *Handbook* discusses standard formats, writing styles, terminology, and examples for both products and services. We are, without doubt, speaking of a functionally useful document.

As with SOWs, DoD has developed clear instructions for specification development. MIL-STD-961E, *Defense and Program-Unique Specifications Format and Content* both directs and assists the practitioner to “identify minimum requirements, list reproducible test methods, allow for a competitive proposal evaluation, and provide for a contract award at the lowest possible cost.” It can be seen that the requirements generated in the JCIDS process must be carefully and exactly translated into the product's specification, and as a matter of convention, those specification requirements “shall be worded such that each paragraph only addresses one requirement or topic.” This point is essential when considering that the requirements in the specification Section 3 must then be identically matched with the verification methods of Section 4. Since the starting point of this article was re-

Relational View of Program Requirements

	ICD CJCISM 3170.01B	CDD CJCISM 3170.01B	WBS MIL-HDBK-245D	System Spec Section 3 MIL-STD-961E	System Spec Section 4 MIL-STD-961E	SOW MIL-HDBK-245D Part III • DAG	TEMP Part IV • DAG
2.X	6.X	1.X	3.X	4.X		3.X	4.X
2.X.X	6.X.X	1.X.X	3.X.X	4.X.X		3.X.X	4.X.X
		1.X.X.1	3.X.X.1	4.X.X.1		3.X.X.1	4.X.X.1
		2.Y				3.Y	
		3.Z				3.Z	

quirements management, limiting requirements to individual paragraphs should also be easier if this edict is followed.

Finally, MIL-HDBK-881A, *Work Breakdown Structures for Defense Materiel Items*, provides an excellent tool for cross checking requirements during program development. In the DoD acquisition context, WBSs are “product-oriented family trees composed of hardware, software, services, data, and facilities” that “relate the elements of work to be accomplished to each other and to the end product.” This definition should not be taken lightly, as it can be easily seen that the definition properly describes a complete system as well as possible component elements. The *Handbook* contains eight specific categories of defense items to be included: aircraft systems; electronic/automated software; missile systems; ordnance systems; sea systems; space systems; surface vehicle systems; and the newest group, unmanned air vehicle systems. These major defense systems can also be combined to define complex composite systems, such as a surface-to-surface missile mounted on a tracked vehicle with both systems containing electronic and computer components. In addition, the *Handbook* provides definitions for the common elements to be considered on any system. Using the handbook as a checklist provides a comprehensive set of considerations that should be addressed on any type system, so rather than having to divine derived requirements out of the ether, the *Handbook* forces the developer to ask whether or not all requirements have been properly addressed. Again, history has provided the user in the field with a powerful aid.

Survival Techniques

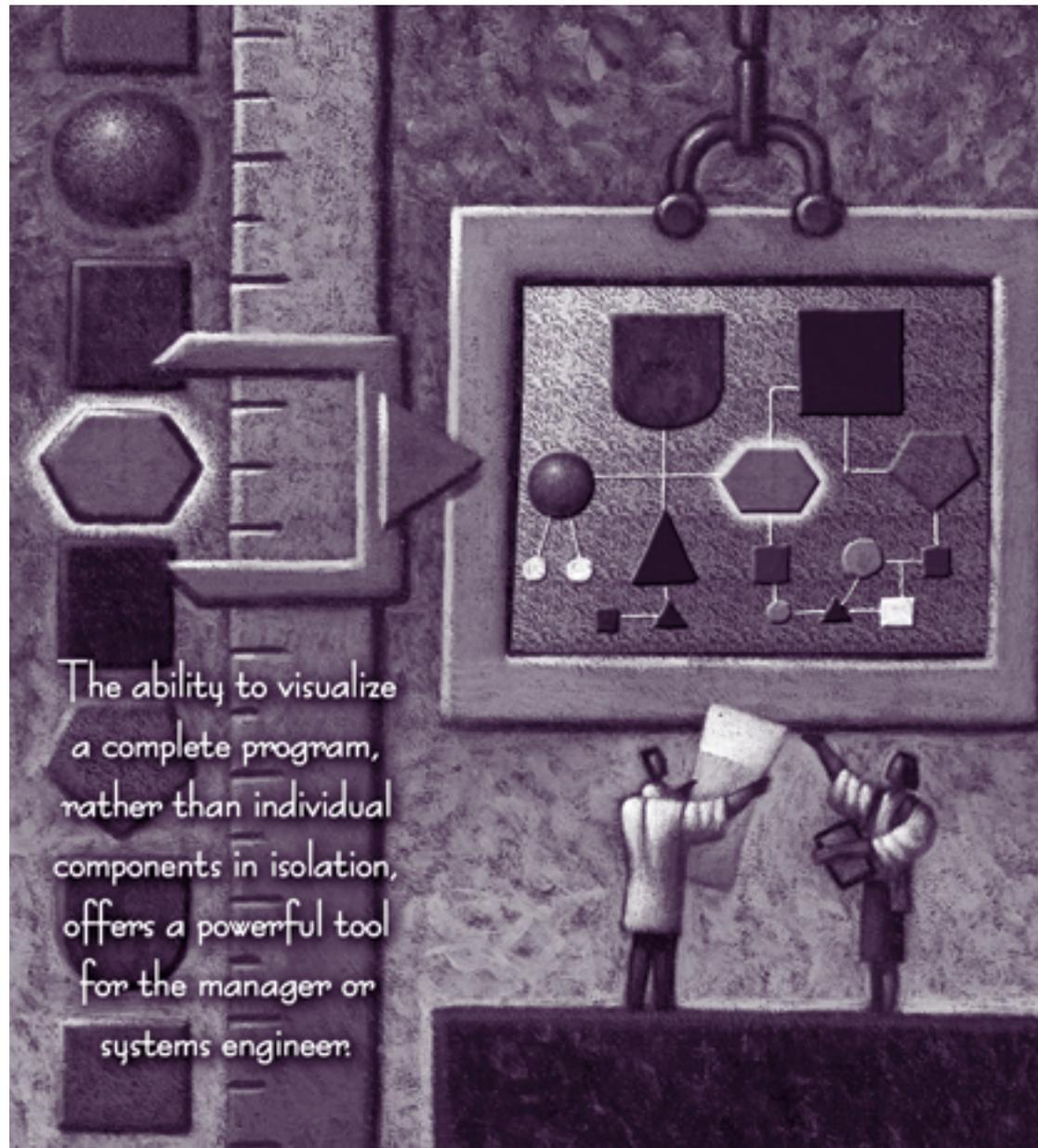
Following this logic process and using the noted tools, we should have made progress in the requirements management process. First, we should now have a reasonable handle on the majority of the requirements definitions; that said, the systems engineering process is both iterative and recursive, so we should expect requirements to change. The point is that we would like to have the vast majority of the requirements identified as early as possible in the program. Second, it is also clear that those requirements must be appropriately integrated into the pro-

gram, including contracting and documentation. That integration effort is the crux of the requirements management process.

One way to view the integration effort is represented graphically on the previous page, illustrating all the elements that have been discussed: ICD, CDD, WBS, specifications, SOW, and finally the Test and Evaluation Master Plan (TEMP). For ease, each column title identifies its source. Missing in the preceding discussion are the relationships among the critical elements, and the requirements in the graphic can now be connected to their output documentation. For example, a concrete Requirement X was identified in both the ICD and CDD, became a portion of the program WBS, and was represented in the systems specification and TEMP. Notice additionally, that in the systems specification, the actual requirement is noted in Section 3 and the verification of that same requirement is noted in

Section 4. Likewise the verification requirements from the systems specification are directly translated into the developmental test (DT) portion of the TEMP, Part III, and translated into the operational test (OT) portion of the TEMP, Part IV. Thus, the graphic allows one to easily visualize the progression of this simplistic requirement to its logical end, and the horizontal progression of the requirement across each row additionally shows where and how every action will be taken.

Likewise, in Requirement X.X, we see that the analysis conducted during the WBS developmental effort has added a new, related Requirement X.X.1. That new requirement is handled in exactly the same manner as Requirement X.X, and this related Requirement X.X.1 is clearly shown in relation to its superior Requirement X.X. In both situations, we can easily visualize the origin of each requirement—Requirements X and X.X from the ICD and CDD,



and Requirement X.X.1 from the WBS. Maintaining these relationships is critical to requirements management.

Requirement Y, like Requirement X.X.1, has its genesis in the WBS effort, but for our example, it is not a performance/technical product. Since only design and performance requirements are hosted in the systems specification, Requirement Y must reside in the SOW. For instance, Requirement Y could be a program management system, delivered by the contractor for inclusion in the overall program master plan. In that same vein, Requirement Z could be a requirement for contractor logistics support and, since it is not a design or performance requirement, should be included in the SOW.

The graphic also demonstrates the proper control and numbering of requirements. In the WBS, we see three related products numbered in series from 1.X through



1.X.X.1; a process (in this case a program management system) numbered beginning with Paragraph 2.Y; and finally another product (in this case contractor logistics support) numbered beginning with Paragraph 3.Y. In the systems specification, numbering of requirements begins in series with Paragraph 3.0, and verification requirements are directly related to the Section 3 requirements, but beginning in an identical series starting with Paragraph 4.0. The TEMP follows that same numbering process. Because of convention, in accordance with MIL-HDBK-245, the only binding SOW requirements are contained in Section 3 and begin in series starting with Paragraph 3.0.

Using this approach meets our first stated criterion (to identify all program requirements), and this approach also partially meets the second and third criteria (to maintain those identified requirements and control any changes). But clearly, this simple manual example would become very cumbersome on a program of any size. To accommodate that sizing problem, a more automated data management approach is required, in this case a relational database. As originally developed by E. F. Codd, a relational database allows the definition of data structures, storage and retrieval operations, and integrity constraints; and these attributes are exactly those required for this task. Using a relational database program thus allows one to automatically fill data fields from one document to another, as well as to maintain configuration man-

agement and configuration history. The addition of a relational database approach fully completes our three initial criteria.

Clear Advantages

This approach to requirements management offers several clear advantages to the practitioner in the field, resulting in improved products for the warfighter. The ability to visualize a complete program, rather than individual components in isolation, offers a powerful tool for the manager or systems engineer. By establishing the relationships between these components, errors can be avoided, and—more important—changes can be understood and managed. In addition to this philosophical approach to requirements management, using available tools such as MIL-STD-961E, or MIL-HDBK-881A or 245D, can simplify the effort to produce well-written program documentation and should be maintained in every acquisition professional's toolbox. Lastly, relational database programs will greatly increase both efficiency and quality on acquisition programs. Combined, these techniques allow professionals to provide higher quality, more cost-effective products to our people in the field.

The author welcomes comments and questions. Contact him at dave.eiband@dau.mil.